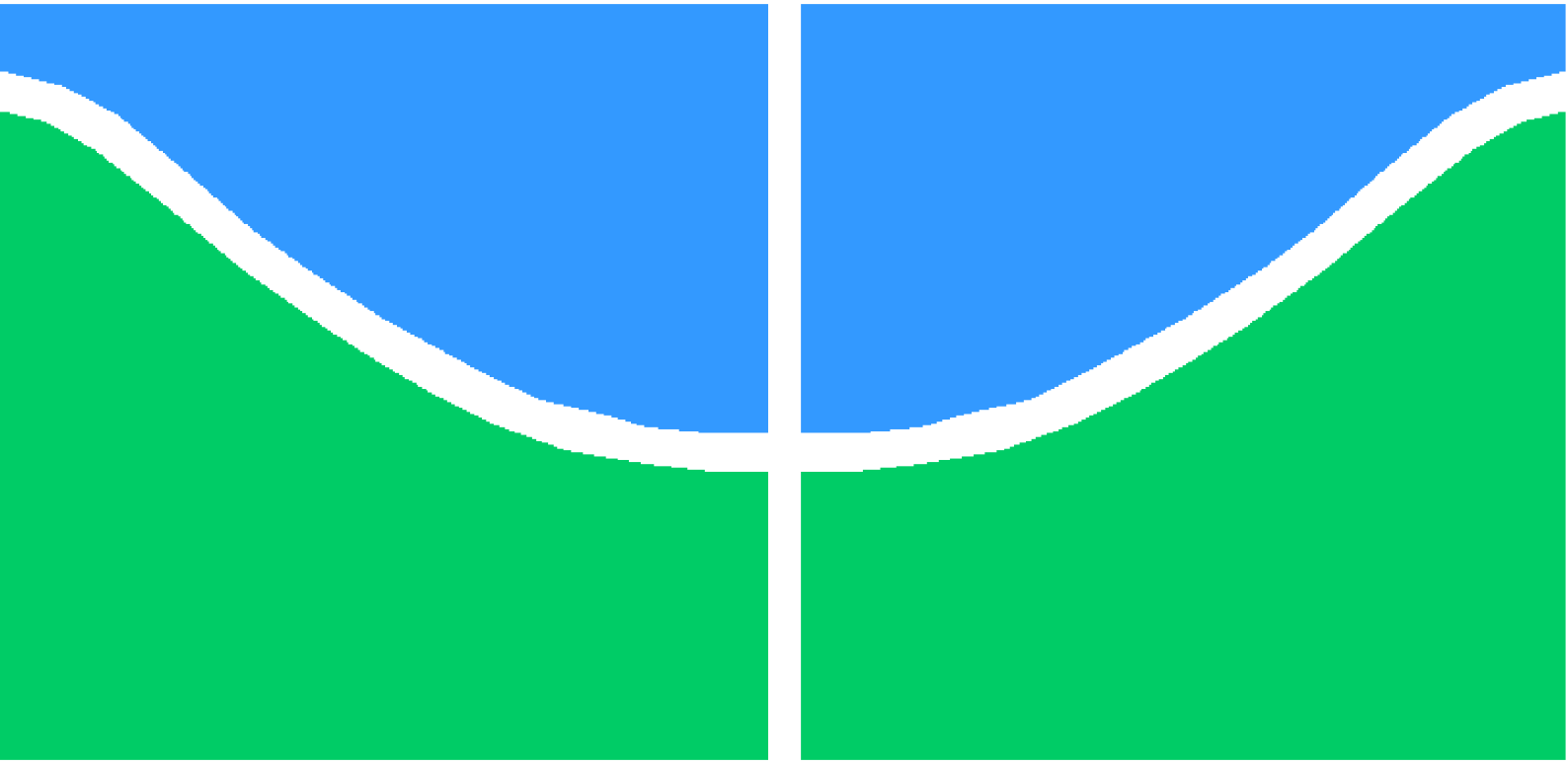

Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia Eletrônica

Sistema modular de medição do movimento humano baseado em ROS e sensores inerciais

Autor: Ana Caroline Alves Amaro
Stefany Santos Aquino

Orientador: Prof. Dr. Roberto de Souza Baptista

Brasília, DF
2019

Ana Caroline Alves Amaro
Stefany Santos Aquino

Sistema modular de medição do movimento humano baseado em ROS e sensores inerciais

Monografia submetida ao curso de graduação em Engenharia Eletrônicada Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Eletrônica.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Prof. Dr. Roberto de Souza Baptista

Brasília, DF
2019

Ana Caroline Alves Amaro
Stefany Santos Aquino
Sistema modular de medição do movimento humano baseado em ROS e sensores inerciais/ Ana Caroline Alves Amaro
Stefany Santos Aquino. – Brasília, DF, 2019-
66 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Roberto de Souza Baptista

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2019.

1. ROS. 2. IMU. I. Prof. Dr. Roberto de Souza Baptista. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Sistema modular de medição do movimento humano baseado em ROS e sensores inerciais

CDU 02:141:005.6

Ana Caroline Alves Amaro
Stefany Santos Aquino

Sistema modular de medição do movimento humano baseado em ROS e sensores inerciais

Monografia submetida ao curso de graduação em Engenharia Eletrônicada Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Eletrônica.

Brasília, DF, 15 de julho de 2019:

Prof. Dr. Roberto de Souza Baptista
Orientador

Prof. Dr.ª Mariana Bernardes
Convidado 1

Prof. Dr.ª Claudia Patricia Ochoa Diaz
Convidado 2

Brasília, DF
2019

Agradecimentos

Gostaria de agradecer a Deus pelo dom da vida, sem ele nada seríamos. Em segundo lugar queria agradecer ao meu pai Luiz Fernandes de Aquino, minha mãe Silvia Madalena dos Santos Aquino e meu irmão Matheus Santos Aquino pelo apoio enorme que sempre tive desde o dia que deixei minha cidade no interior de Goiás para vir para Brasília. Sempre me dando forças para não desistir mesmo com as dificuldades e, principalmente, a distância. Agradecer ao meu namorado Marcos Vinicius Abdalla por me dar forças todo momento, ser minha calmaria em meio ao caos e sempre acreditar no meu potencial. Agradecer a segunda família que Deus me deu em Brasília, Larissa, Marcus Vinicius, João Paulo e Diana que sempre estão presente na correria da vida. Agradecer a minha dupla Ana Caroline pelo apoio e parceria nesse trabalho. Por fim, não menos importante, à todos meus amigos de Ipameri que por mais que a distância seja grande nunca me abandonaram.

Agradeço primeiramente à Deus que iluminou o meu caminho durante essa caminhada. Agradeço também, aos meus pais, Maria Aparecida Alves da Cruz e Gláucio Rogério de Faria Amaro que nunca mediram esforços para que hoje eu estivesse aqui. Ao meu irmão Gláucio Felipe Alves Amaro que sempre se fez presente. E ao meu namorado, Kaleb Estanislau, que está comigo desde o início desta jornada, obrigado pelo carinho, amor e paciência.E também ao meu amigo e colega de faculdade, Gabriel Rolim Moreira, por tantas alegrias compartilhadas.

“ Educação é uma descoberta progressiva de nossa própria ignorância.”

François-Marie Arouet - Voltaire

Resumo

O presente trabalho apresenta uma proposta de um sistema modular de medição do movimento humano baseado em ROS e sensores inerciais. O objetivo do projeto é projetar um protótipo de um sistema de medição dos ângulos da articulação do cotovelo utilizando duas IMUs, uma localizada no braço, próximo ao cotovelo e outra localizada no antebraço. O ROS facilita a interconexão entre as unidades de medição inercial possibilitando o acoplamento de múltiplas IMUs. O projeto final apresenta um sistema capaz de captar os ângulos da junta do braço através de dois MPU-6050 acoplados ao corpo que realizam a aquisição desses dados que são tratados e expostos como uma subtração das angulações do movimento de inclinação pitch e roll.

Palavras-chaves: IMU. ROS. Sensores Incerciais.

Abstract

The present work presents a proposal of a modular system of human movement measurement based on ROS and inertial sensors. The objective of the project is design a prototype of an elbow joint angle measurement system using two IMUs, one located in the arm, near the elbow and another located on the forearm. The ROS facilitates the interconnection between the inertial measurement units allowing the coupling of multiple IMUs. The final design features a system capable of capturing the arm joint angles through two MPU-6050 coupled to the body that perform the acquisition of these data that are treated and exposed as a subtraction of the angles of pitch and roll tilt movement.

Key-words: IMU. ROS. Inertial Sensors.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

PCI	Placa de Circuito Impresso
CPU	Central Processing Unit
MEMS	Microeletomechanical Systems
DOF	Degree of Freedom
IMU	Inertial Mensurement Unit
CI	Circuito Integrado
I2C	Inter-Integrate Circuit
SLC	Serial Clock
SDA	Serial Data
MPU	Motion Process Unit
ROS	Robotic Operating System
OSRF	Open Source Robotics Foundation
OS	Operacional System
PCL	Point Cloud Library
RTT	Real Time Text
GPIO	General Purpose Input / Output
RPi	Raspberry Pi
LTS	Long Term Support
IBM	International Business Machines
AMD	Advanced Micro Devices

Lista de símbolos

β	Letra grega Beta
γ	Letra grega Gamma
α	Letra grega minúscula alpha

Sumário

1 Introdução

1.1 Contextualização

A análise de movimento humano é uma área importante no campo médico, a qual é frequentemente necessária para diagnóstico médico e na fisioterapia (ONG et al., 2018a). Diferentes aplicações em tecnologias assistivas e reabilitação necessitam de uma avaliação quantitativa da tarefa executada, como a medição do movimento articular (BO; HAYASHIBE; POIGNET, 2011).

O rastreamento preciso do movimento dos membros sempre foi um requisito crucial para inúmeras áreas, como telemedicina, reabilitação ortopédica e neurológica, jogos interativos e realidade virtual (BUONOCUNTO; MARINONI, 2014).Várias tecnologias e abordagens estão disponíveis para produzir sistemas de rastreamento de movimento, que derivam de estimativas de orientação. Uma abordagem cada vez mais popular é embasada no uso de sensores inerciais. As unidades de medição sensoriais(IMU's), compostas de sensores inerciais, estão tornando-se mais populares em vários ramos, incluindo a robótica na biomédica.

Sensores inerciais vestíveis são adequados para medir movimentos de alta frequência com baixa amplitude, características difíceis de serem medidas em exames de imagem (FONG et al., 2004). Um dispositivo hábil, de simples manuseio e baixo custo foi desenvolvido em uma plataforma embarcada, visando um sistema prático e eficiente.

1.2 Relevância do Trabalho

Usando as orientações de segmentos de um corpo de um indivíduo, o conhecimento sobre o comprimento e as características da junta, posições relativas no corpo e ângulos entre segmentos podem ser estimados. As medidas de posição do corpo como, por exemplo, ângulos de juntas, são necessárias em muitas aplicações, como em tecnologias assistivas com o propósito de estimular os pacientes.

Propõe-se um sistema inovador, capaz de auxiliar na recuperação de movimentos perdidos do membro superior, por exemplo, melhorando a qualidade das atividades realizadas no dia-a-dia de um portador de necessidades especiais, trazendo mais autossuficiência a sua vida. O sistema criado é composto de um dispositivo vestível que tem potencial para ser utilizado em diferentes aplicações, desde treinamento esportivo à reabilitação, permitindo a aquisição de dados do membro superior a fim de estimar o desempenho dos movimentos do mesmo.

1.3 Motivações

A capacidade de se mover é uma condição básica para interagir com o meio ambiente. As restrições de movimento levam à perda da qualidade de vida e à incapacidade de viver uma vida autônoma e independente. A fisioterapia baseada no treinamento repetitivo beneficia-se da prerrogativa da plasticidade do cérebro, que permite a recuperação dos movimentos. A utilização de ferramentas robóticas em ambiente hospitalar, atua como uma abordagem para acelerar o processo de reabilitação, maximizando a intensidade de treinamento dos pacientes sendo uma poderosa ferramenta de auxílio no âmbito da fisioterapia. A robótica de reabilitação deve ser expandida e melhorada todos os dias, a fim de ajudar inúmeras pessoas. Possibilita entre muitos benefícios, a reprodução de treinamentos, visando restaurar funções perdidas por traumas, doenças crônicas.

Os sensores inerciais, não dependem de referência externa, e são capazes de medir quantidades físicas, tais como aceleração linear e velocidade angular, que estão relacionadas com o movimento de objetos onde os sensores são fixados. Avanços tecnológicos recentes no campo de sistemas microeletromecânicos, tornaram possível fabricar sensores inerciais de custo relativamente baixo, altamente miniaturizados e com consumo de energia limitado. Sistemas de sensores inerciais estão disponíveis no mercado a um baixo custo e são eficazes. Os sensores inerciais mais utilizados são acelerômetros e giroscópios.

1.4 Objetivos

Neste trabalho é apresentado um sistema robótico que utiliza o sistema operacional para robôs (ROS) para integração das unidades de medição inercial (IMUs). Com intuito de atuar em sistemas de tecnologias assistivas, propõe-se esse dispositivo capaz de auxiliar no monitoramento do movimento humano voltado para o contexto de reabilitação.

Sensores inerciais podem ser amostrados em alta taxas, e requerem pouca energia. Acelerômetros e giroscópios pertencem a tecnologia citada, medem as mudanças de posição e orientação. Combinando medições de ambos os sistemas, estima-se que seja uma solução eficiente para estimativas de posição e orientação. Embora os sistemas de sensores inerciais proporcionem uma configuração apta a finalidade proposta, ainda há uma série de desafios à frente relacionados à integração de sistemas e estimação de ângulos.

2 Revisão Bibliográfica

2.1 A tecnologia e o movimento humano

O movimento humano é complexo e variado, sofre mudanças significativas ao longo da vida de cada indivíduo. Da mesma forma que as capacidades físicas mudam, a qualidade do movimento também muda. Essas mudanças podem ser atribuídas ao envelhecimento, progressão de doença, reabilitação, ou melhorias no movimento devido a esportes e outro treino (KULIC; VENTURE; NAKAMURA, 2009).

No campo da medicina, a tecnologia é uma forte aliada amplamente utilizada em ortopedia, cirurgias, fisioterapia e outras áreas para ajudar os médicos a compreenderem inteiramente a condição dos pacientes por meio da comparação dos movimento dos pacientes antes e após a reabilitação (CHEN et al., 2015).

Há várias situações, nas quais o conhecimento de características precisas do movimento dos seres humanos são demasiadamente úteis, impactando de modo direto no diagnóstico, monitoramento e compreensão de patologias, entre outros. No âmbito esportivo, o monitoramento em tempo real do estado de treinamento dos atletas é realizado por meio de uma interface de rastreamento nos atletas que utilizam equipamentos de captura de movimento, como coletes com sensores vestíveis (XU, 2013).

Os esportistas precisam ser monitorados para melhorar o desempenho esportivo ou corrigir problemas que podem causar danos futuros de acordo com Milosevic e Farella (2015). Vale ressaltar a relevância na Terapia Ocupacional também, onde profissionais de TO conhecendo as características do movimento das pessoas, avaliarão com mais propriedade problemas ligados ao trabalho (HONDORI; KHADEMI; LOPES, 2012).

Como na medicina tradicional, na medicina veterinária, pode ser usado para reabilitação e avaliação do movimentos de animais (VENKATRAMAN et al., 2007). É de suma importância em diversas áreas um mecanismo apto a aferir a amplitude do movimento articular do corpo humano (CALLEJAS-CUERVO; ALVAREZ; ÁLVAREZ, 2016).

Modelos cinemáticos de corpos rígidos articulados são necessários na análise e controle de muitos sistemas de engenharia. Satélites, robôs e ligações mecânicas são alguns exemplos de sistemas que fornecem o movimento controlado de corpos rígidos no espaço para realizar uma variedade de tarefas (HEMAMI, 1982).

2.2 Sistemas inerciais

Existem diversos sistemas que capturam o movimento do corpo humano, essas tecnologias estão sendo aperfeiçoadas ao longo dos anos. Gypsy (2016) teve como estudo o rastreamento através de sistemas eletromecânicos, já Qualisys (2016) teve estudos baseados em rastreamento visual. Liberty (2016) teve todo seu trabalho baseado em sistemas de rastremaento magnético.

Várias são as desvantagens associadas a esses sistemas como: alto custo, osbtrução dos marcados em sistemas ultrassônicos e ópticos, alta imprecisão em sistemas mecânicos. Sistemas baseados em rastremaneto inercial são simples, possuem valor acessível e possui diversas aplicações na área da saúde (XSENS, 2016).

Os sensores inerciais são acoplados à membros do corpo para realizar o rastreamento dos movimentos e possibilitar a representação de sistemas por ângulos. Essas representações são muito utilizadas em sistemas robóticos e encontram desafios para estimar a melhor posição desses sensores no corpo. Definir a melhor posição tem impacto na taxa de ruídos relacionados à interferência do sinal (LIN, 2012).

A estimativa dos ângulos medidos pelos sensores tem sido feita por uma orientação inicial combinada com a velocidade angular, que é medida pelo giroscópio. Da mesma forma, essa medida pode ser realizada pela integração dupla da aceleração medida pelo acelerômetro. Porém, existe um problema relacionado com essa integração de sinais. Como o giroscópio possui um drift relacionado ao seu sistema interno e o aceleômetro é muito sensível a qualquer perturbação isso faz com que um erro acumulativos seja introduzido ao sinal de saída, necessitando de um sistema externo para correção (FOXLIN, 2005).

O sucesso dos algoritmos para sistemas inerciais possui limitações. As estimativas são de confiança na medida em que a única aceleração resultante seja a aceleração da gravidade. Movimentos complexos do corpo humano necessitam de uma grande quantidade de unidade de medidas inerciais, este é outro fator de limitação, já que quanto maior o número de IMUs mais equações não lineares compõem o modelo, mais dimensões são acrescentadas e isso torna o algoritmo caro e instável (MADGWICK, 2010).

2.3 Tecnologia MEMS

A engenharia evolutiva vem juntando esforços para reduzir o tamanho de sistemas diversificados, ao mesmo tempo tentam manter seu poder e aumentar seu desempenho. Os sistemas microeletromecânicos (MEMS) representam um esforço radical de transformar a escala, desempenho e custo desses sistemas, empregando técnicas de microprocessos que permitem a integração de sistemas mecânicos e sensores em apenas um chip. A tecnologia MEMS permitiu vários tipos de sensores, atuadores e sistema a ser reduzido em tamanho

por ordens de magnitude, enquanto muitas vezes até melhora o desempenho do sensor (JUDY, 2001).Esta tecnologia visa um sistema independente de detecção e atuação inter relacionando dispositivos em conjunto com processamento de sinal e controle eletrônico em um único substrato (POLLA, 1996).

Existem muitas técnicas atualmente sendo utilizadas na produção de diferentes tipos de MEMS, incluindo micro sensores inerciais, que tornaram possível fabricar MEMS em grandes volumes a baixo custo individual. Giroscópios e acelerômetros são considerados dispositivos da tecnologia MEMS e têm um grande potencial para diferentes tipos de aplicações como sensores primários de informação para sistemas de orientação, controle e navegação. Eles representam uma importante tecnologia inercial porque outros dispositivos, com a mesma funcionalidade, não permitem uma miniaturização significativa. Os sensores MEMS são comumente aceitos como sensores de baixo desempenho e baixo custo (APOSTOLYUK, 2006).

2.4 IMU

A unidade de medida inercial (IMU) consiste em duas tríades sensoriais ortogonais, uma composta por três acelerômetros, o outro de três giroscópios. Nominalmente, os eixos das duas tríades são paralelos e a origem é definida como a origem da tríade do acelerômetro. Os eixos do sensor são fixos no corpo do IMU e são chamados de eixos do corpo. Eles formam a armação do corpo. A primeira tríade do sensor produz uma força específica ao longo dos três eixos da estrutura; a segunda tríade gera uma taxa angular sobre os mesmos eixos. Ambos os conjuntos de medições são feitos em relação a um referencial inercial. O posicionamento inercial, portanto, baseia-se no princípio simples de que as diferenças na posição podem ser determinadas por uma integração dupla de aceleração, sentida em função do tempo, em um quadro de coordenadas estável e bem definido (EL-SHEIMY S. NASSAR, 2004).

Na última década, os sensores inerciais se tornaram um procedimento comum para mensurar o movimento humano. Os giroscópios presentes na IMU, fornecem uma medida precisa de velocidade angular em torno de um certo eixo, mas pequenos erros de velocidade resultam em um desvio de orientação. Sob condições de aceleração não linear, os dados do acelerômetro permitem a determinação da inclinação do sensor em relação à gravidade, que pode corrigir o desvio (LAMBRECHT; KIRSCH, 2014).Esses sensores são baseados na tecnologia MEMS e são utilizados para diversas aplicações como rastreamento e medições biomecânicas. Não funcionam bem como rastreadores inerciais sem calibração. É necessário sistemas para remover os erros sistemáticos desses sensores (FONG W. T. ; ONG, 2008).

Sensores vestíveis tem sido muito usados para medir o movimento humano, tam-

bém podem ser usados para monitorar os aspectos cinemáticos do desempenho (JACOB; ZAKARIA; TOMARI, 2016).Unidades de medição inercial (IMU's) fornecem dados de orientação e aceleração de um corpo em movimento, e são extensamente usadas na área da biomédica[14]Sensores inerciais são de fácil acesso, baixo custo e tamanho compacto. Muito utilizados para estimar ângulos dos membros humanos(WILLEMSEN; FRIGO; BOOM, 1991).

IMUs também podem ser combinadas com sensores de posição para formar rastreadores híbridos de seis graus de liberdade (6DOF), que podem medir a posição e a orientação. Exemplos de rastreadores híbridos 6DOF com IMUs são apresentados por Titterton D. ; Weston (1997), Foxlin (2005) e Niu X. ; El-Sheimy (2005).

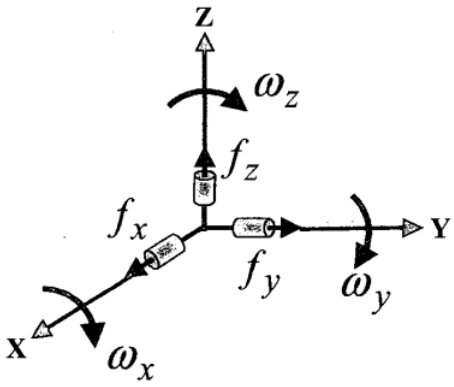

Figura 1 – Exemplificação da medição do acelerômetro e giroscópio.(Adaptado de (EL-SHEIMY S. NASSAR, 2004).

O módulo utilizado neste trabalho, GY-521, contém um único CI o MPU6050, uma IMU composta de acelerômetros e giroscópio, além de um sensor de temperatura que não será utilizado no presente projeto. Este módulo possui uma vantagem por possuir uma biblioteca chamada Motion Processing Unit (MPU) que consegue integrar os dados dos sensores e processar o algoritmo de detecção do movimento no próprio módulo, livrando o microcontrolador desta tarefa. Isso é útil pois libera a energia do processador raspberry, que tem um potencial menor .Além disso, o MPU6050 possui conversores digitais/analógicos que apresentam uma resolução de 16 bits para cada canal, um do acelerômetro e outro do giroscópio. Essa resolução para cada canal permite que os dados provindos dos sensores sejam amostrados ao mesmo tempo.(CALACHE, 2013)

Figura 2 – CI MPU6050 que compõe o módulo GY-521. Adaptado de (ARDUINO, 2018).

A figura abaixo representa o módulo utilizado, GY-521 conectado ao micrcontrolador Raspberry. Para este projeto a comunicação utilizada entre o microcontrolador,Raspberry Pi, e o sensor MPU6050 foi o protocolo de comunicação I2C que será discutido posteriormente.

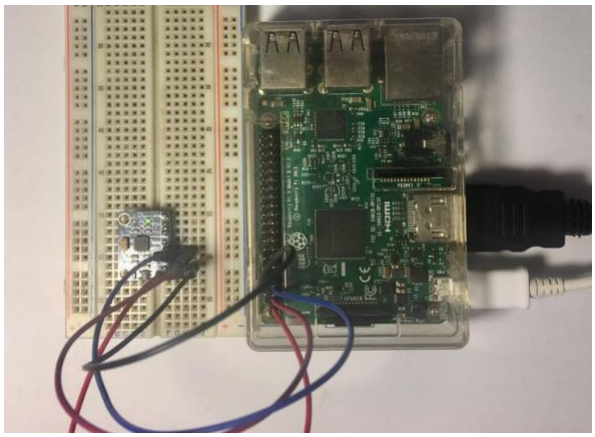

Figura 3 – Módulo GY-521 conectado ao Raspberry Pi (Autoria própria)

Os ângulos podem ser calculados a partir dos dados obtidos do acelerômetro, juntamente com as velocidades angulares recebidas e podem ser usados por exemplo, para a análise de marcha dos membros inferiores (VILLENEUVE et al., 2016).

Em Yin et al. (2013), um sistema sem fio foi desenvolvido usando sensores inerciais magnéticos e de baixo custo. Os quais, colaboraram na análise do processo da "tacada de golfe (golf swing)". No decorrer da experiência, confirmaram que o sensor de movimento (MPU-6050) era capaz de capturar a aceleração máxima durante a tacada.

Em Anuar, Sahari e Yue (2016), desenvolveu-se um protótipo baseado na problemática que um sistema de rastreamento do movimento do corpo fornece valores de alta precisão, exatidão e confiabilidade, comparando-se com uma avaliação feita por outro ser humano. Dois módulos GY-521 são usados para captar os movimentos dos membros humanos, respectivamente um ligado na parte superior do braço e o outro no antebraço, assimilando a rotação ao longo dos eixos x ,y e z. Assim, realizando a captura do ângulo e a orientação do membro do usuário e enviando ao computador via Arduino.

Os dados brutos do sensor serão filtrados primeiro antes de usar e os valores filtrados serão enviados para um programa de visualização 3D onde o movimento do membro do usuário pode ser visualizado instantaneamente. Semelhante ao que foi projetado em Ong et al. (2018b), onde um sistema com múltiplas IMU's capaz de exibir o movimento do membro superior de uma pessoa em tempo real, mostrando dados de aceleração, velocidade angular, posição e orientação extraídos e diretamente expostos. A medição do potencial mioelétrico foi realizada em Seo, Noh e Jeong (2018), usando sensores inerciais. Os dados físicos obtidos com o giroscópio e acelerômetro, e as informações do eletromio-

grama, são enviados para análise e processamento. Implementaram um algoritmo capaz de distinguir o momento e relacionar com a postura, para fornecer informações da postura e maximizarem a eficiência dos exercícios recomendados durante a reabilitação.

No campo clínico, a análise de marcha (gait) é importante para avaliar quantitativamente os padrões da marcha humana e quantificar deficiências (WREN et al., 2011). Em Gastaldi et al. (2016), criou-se um sistema com sensores inerciais vestíveis que permitem medir e registrar a marcha em condições naturais para o paciente. Em [50], os dados obtidos a partir do sensor MPU-6050 do acelerômetro X, Y e eixo Z foram denominados como (Ax), (Ay) e (Az). Enquanto os dados giroscópio foram denominados como (Gx), (Gy) e (Gz). Com base na escala usada, usa-se os dados brutos do acelerômetro na seguinte fórmula:

$$A_{eixo} = \frac{escala}{A_{DadosPuros}} \tag{2.1}$$

A_{eixo} é o eixo que será calculado por exemplo Ax, Ay e Az. Os dados eixo puros são os valores puros de x, y e z do acelerômetro. Utiliza-se a seguinte expressão para os dados do giroscópio:

$$G_{eixo} = \frac{escala}{G_{DadosPuros}} \tag{2.2}$$

Assim, G_{eixo} , é o eixo que será calculado Gx, Gy e Gz. E $G_{DadosPuros}$ são os valores brutos recebidos do giroscópio. Para se calcular o valor de magnitude dos dados do eixo xyz no acelerômetro, utiliza-se:

$$A_T = \sqrt{aX^2 + aY^2 + aZ^2} \tag{2.3}$$

Para a análise do movimento das juntas do braço desprezaremos um eixo e analisaremos um vetor nos eixos X e Y. Os ângulos provindos de cada IMU serão referenciados separadamente em relação a terra e para adquirir um ângulo de posição final será realizado a subtração dos ângulos provenientes de cada sensor.

2.5 Comunicação I2C

O barramento I2C foi desenvolvido em 1982. Sua finalidade original era fornecer uma maneira simples de conectar uma CPU a chips periféricos em um aparelho de televisão. Dispositivos periféricos em sistemas embarcados são freqüentemente conectados ao microcontrolador .Uma maneira comum de fazer isso é conectar os periféricos ao endereço paralelo do microcontrolador e aos barramentos de dados. Isso resulta em muita fiação na placa de circuito impresso (PCI) e requer uma certa lógica para interpretar o barramento

de endereços no qual todos os periféricos estão conectados. A fim de poupar os pinos do microcontrolador, uma lógica adicional e simples, foi elaborada nos laboratórios da Philips em Eindhoven, Holanda, criaram o protocolo I2C que só requer dois fios para conectar todos os periféricos a um microcontrolador (GRECU; IORDACHE, 2015).

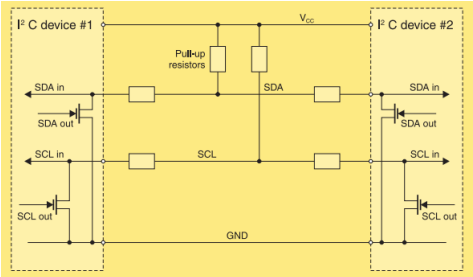

Figura 4 – Protocolo I2C com dois dispositivos conectados. Adaptado de (LEENS, 2009).

O I2C é um protocolo multi-mestre que usa duas linhas de sinal. Os dois sinais I2C são chamados de dados em série (SDA - Serial Data) e sinal de relógio serial (SCL - Serial Clock). Não há necessidade de selecionar chip (seleção de escravos) ou lógica de arbitragem. Praticamente qualquer número de escravos e qualquer número de mestres podem ser conectados a estas duas linhas de sinal e assim, serão capazes de se comunicar usando um protocolo que define inicialmente que um sinal de clock (SCLK) enviado do barramento mestre para todos os escravos, todos os sinais SPI estão sincronizados com este sinal de clock; um sinal de seleção escravo (SSn) para cada escravo, é utilizado para selecionar o escravo que o mestre comunica uma linha de dados do mestre para os escravos, denominada Master Out-Slave In (MOSI) e uma linha de dados dos escravos para o mestre, denominada Master In-Slave Out (MISO).

Fisicamente, o barramento I2C consiste dos dois fios ativos SDA e SCL e uma conexão terra. Os fios ativos são bidirecionais. A especificação do protocolo I2C afirma que o IC que inicia uma transferência de dados no barramento é considerado o mestre de barramento. Consequentemente, naquele momento, todos os outros CIs são considerados do barramento escravo. A especificação I2C afirma que os dados só podem mudar na linha SDA se o sinal do SCL estiver em nível baixo; inversamente, os dados na linha SDA são considerados estáveis quando o SCL está em estado alto (GRECU; IORDACHE, 2015).

O SPI é um protocolo de comunicação de um único mestre. Isso significa que um dispositivo central inicia todas as comunicações com os escravos. Quando o mestre SPI deseja enviar dados para um escravo e / ou solicitar informações a partir dele, ele seleciona um escravo puxando a linha SS correspondente para baixo, e ativa o SCL em uma freqüência de clock utilizável pelo mestre e escravo. O mestre gera informações na linha MOSI enquanto faz a amostragem da linha MISO (SZEKACS; SZAKAILL; HEGYKOZI, 2007). O I2C e o SPI são adequados para comunicações entre circuitos integrados para

velocidade de transferência de dados baixa /média com periféricos integrados (OUDJIDA et al., 2009).

2.6 Raspberry PI

Devido a grandes avanços em toda a esfera da informática,e com a grande demanda por dispositivos embarcados alimentados por bateria, acarretou-se numa diminuição cada vez mais notável no tamanho do hardware e no aumento da eficiência, poupando mais energia a cada versão lançada, destaca-se por seu significativo poder de processamento comparado ao seu tamanho. A Raspberry Pi é um mini computador com dimensões de um cartão de crédito, capaz de se conectar a monitores de Tv/computador, manuseado com periféricos comuns, basta acoplar o teclado e mouse a uma das saídas disponíveis.Ela foi escolhida para o trabalho, primeiramente pelo seu baixo custo e por sanar a necessidade de uma placa de pequenas dimensões que tornasse viável nosso sistema modular de medição do movimento humano.

A Fundação Raspberry Pi, fornece distribuições como, Debian e o Arch Linux ARM e também Python como a linguagem de programação principal. Python foi escolhido como o principal linguagem de programação, como geralmente é aceito, fácil de aprender e uma linguagem de programação completa adequada para aplicações no mundo real. Devido às vantagens únicas do sistema Raspberry Pi, esta tecnologia é muito promissora para fornecer soluções no mundo em desenvolvimento.A mais distinta característica do Raspberry Pi quando usado para fins educacionais é o módulo GPIO, que permite a interface com o propósito geral da eletrônica (GOYAL, 2014).

Em 2006, os primeiros conceitos da Raspberry Pi (RPi) foram baseados no microcontrolador Atmel ATmega644, onde os idealizadores do projeto, pensaram em um jeito de interromper o declínio gradual do interesse dos estudantes pela área da Computação no Reino Unido (MORENO, 2016). Foi desenvolvido em 2012 no Laboratório de Computação da Universidade de Cambridge, Reino Unido, pela Fundação Raspberry Pi, cuja finalidade principal é promover a disseminação do ensino de Ciências da Computação Básica em escolas por todo o mundo. Disponível no mercado desde 29 de fevereiro de 2012, teve preço inicial de 35 dólares (CAMERON, 2013).

Diante da notável necessidade de um sistema móvel que fosse capaz de carregar e executar a ROS com eficácia, preço acessível, e de fácil manipulação, alguns modelos da plataforma encaixam-se nas especificações. As principais versões compatíveis com o sistema ROS, são: Raspberry Pi 2, RPi 3 e RPi 3 B. A RPi 3 B+ não oferece suporte ao sistema Ubuntu Mate, requerido atualmente para o pleno funcionamento de todas as aplicações do sistema.O desempenho da RPi requer um sistema operacional embarcado, encarregado de toda a coordenação dos recursos e funcionalidades da plataforma.

2.7 Sistemas Operacionais para plataformas embarcadas

O Ubuntu é um dos maiores sistemas operacionais de desktop baseados em Linux do mundo. O Linux está no coração do Ubuntu e possibilita a criação de sistemas operacionais seguros, poderosos e versáteis, como o Ubuntu e o Android. O Android está agora nas mãos de bilhões de pessoas em todo o mundo e também é alimentado pelo Linux. Ao contrário do Windows e do OS X, o Linux não é criado e suportado por apenas uma empresa. É suportado pela Intel, pela Redhat, pela Linaro, pela Samsung, pela IBM, pela SUSE, pela Texas Instruments, pela Google, pela Canonical, pela Oracle, pela AMD e pela Microsoft. Mais de 4.000 desenvolvedores contribuíram para o Linux nos últimos 15 anos (UBUNTU, 2018).

Dois sistemas operacionais podem ser destacados entre os suportados pelo ROS. O Raspbian, que é um sistema operacional baseado em Debian, uma distribuição do Linux, projetado para Raspberry Pi.Porém não oferece todas as bibliotecas necessárias, nem o suporte ao sistema ROS Kinetic. E dentre os sistemas suportados pela RPi 3B, escolhida para o projeto, encontra-se o Ubuntu Mate. Particularmente desenvolvido para atuar em plataformas RPi 2 , 3 e 3B, já citada anteriormente. O Ubuntu MATE carrega o sistema operacional básico do Ubuntu e introduz o MATE Desktop,este último, responsável por prover as interfaces gráficas essenciais para controle e total funcionalidade de um computador, enquanto o Ubuntu MATE acrescenta diversos aplicativos e recursos capazes de tornar seu computador uma estação de trabalho de alto desempenho e eficácia. O Ubuntu MATE é uma distribuição Linux que visa trazer a simplicidade e elegância do sistema operacional Ubuntu através de um ambiente de desktop clássico e tradicional o desktop MATE (SMITH, 2015).

2.8 Ângulos de Euler

A orientação relativa de dois sistemas de coordenadas ortogonais podem ser especificados por, um conjunto de não menos que três ângulos, geralmente chamados de ângulos de Euler, são popularmente utilizados em análises de dinâmica de corpos rígidos. Porém, mesmo o conceito sendo universalmente conhecido, não há uma definição comum. Há um grande número de escolhas possíveis para os três ângulos exigidos para definir um conjunto de ângulos de Euler. A ideia básica afirmada por Euler é que a orientação relativa de dois sistemas de coordenadas pode ser especificada por um conjunto de três ângulos independentes (PIO, 1996)

Em Ang e Tourassis (1987), observa-se que a descrição da posição e orientação de um corpo rígido em relação a um referencial cartesiano $R = \{O_R; X_R; Y_R; Z_R\}$ é de extrema importância na cinemática. Para descrever o movimento de um corpo rígido, um referencial cartesiano $E = \{O_E; X_E; Y_E; Z_E\}$ é relacionado a ele. Primeiramente, a posição

do corpo rígido é definida pelo vetor de posição $(O_R; O_E)$ da origem O_E da estrutura do corpo rígido; e logo após, o do corpo rígido é definido pela matriz de rotação $R = \{X_E; Y_E; Z_E\}$, onde os vetores unitários X_E, Y_E e Z_E descrevem os eixos da estrutura do corpo rígido.

$$(O_R; O_E) = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}$$

$$(X_E; Y_E; Z_E) = \begin{bmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{bmatrix}$$

Assume-se que $O_R = O_E \Leftrightarrow P_x = P_y = P_z = 0$.

A abordagem da matriz de rotação utiliza nove parâmetros (que obedecem a restrições de ortogonalidade e comprimento de unidade) para descrever a orientação de um corpo rígido. A orientação é então definida por uma rotação finita s sobre o eixo $k = (k_x, k_y, k_z)^T$.

Na representação de Euler, é necessária uma série de três rotações ordenadas para a direita para coalescer o referencial com a estrutura do corpo rígido. Especificamente, o referencial é girado inicialmente em $Z_R = [0 \quad 0 \quad 1]^T$ sobre o eixo por um ângulo α , para que o plano (X_R, Z_R) contenha Z_E. Então, sobre o novo eixo $Y_R = [-\sin\alpha \quad \cos\alpha \quad \mathrm{O}]^T$ por um ângulo β de modo que $Z_R = Z_E$. Assim, sobre o novo eixo $Z_R = [\cos\alpha \sin\beta \quad \sin\alpha \sin\beta \quad \cos\beta]^T$ por um ângulo γ de modo que $X_R = X_E$.

$$Euler(\alpha, \beta, \gamma) = \begin{bmatrix} \cos\alpha\cos\beta\cos\gamma - \sin\alpha\sin\gamma & -\cos\alpha\cos\beta\sin\gamma - \sin\alpha\sin\gamma & \cos\alpha\sin\beta \\ \sin\alpha\cos\beta\cos\gamma - \cos\alpha\sin\gamma & -\sin\alpha\cos\beta\sin\gamma - \cos\alpha\cos\gamma & \sin\alpha\sin\beta \\ -\sin\beta\cos\gamma & \sin\beta\sin\gamma & \cos\beta \end{bmatrix}$$

A orientação do corpo rígido é completamente especificada pelos três ângulos de Euler, α, β, γ. A transformação inversa pode ser calculada prontamente a partir da matriz de rotação, observando que:

$$a_x = \cos\alpha; a_y = \sin\alpha\sin\beta; a_z = \cos\beta\sin\beta \tag{2.4}$$

$$n_z = -\sin\beta\cos\gamma; o_z = \sin\beta\sin\gamma \tag{2.5}$$

Se $sin\beta \neq 0 \Leftrightarrow |a_x| + |a_y| \neq 0$, o problema de transformação inversa tem duas soluções (α, β, γ) e $(\alpha + 180^o, -\beta, \gamma + 180^o)$, onde:

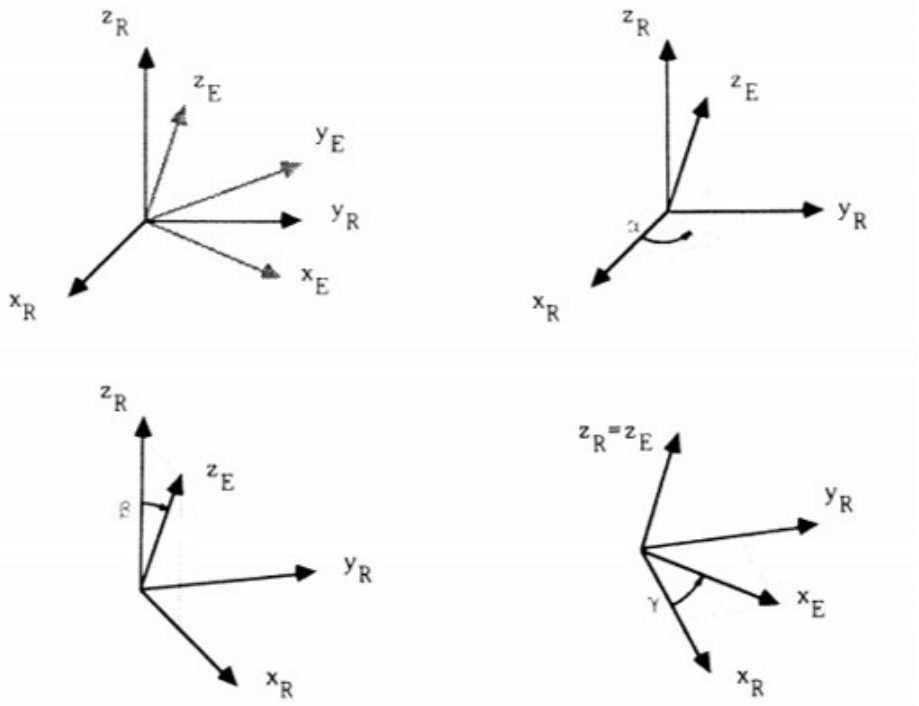

Figura 5 – Definição ângulos de Euler(ANG; TOURASSIS, 1987).

$$\alpha = \arctan 2(a_x, a_y)$$

$$\beta = \arctan 2 + (\sqrt{a_x^2 + a_y^2 + a_z^2})$$

$$\gamma = \arctan 2(o_z, -n_z)$$

As singularidades matemáticas ocorrem nas interseções entre múltiplas regiões de solução para o problema de transformação inversa. Essas singularidades significam orientações no espaço, onde taxas rotacionais infinitas são necessárias para produzir uma velocidade angular finita e aceleração do corpo rígido. As singularidades matemáticas podem surgir dependendo da representação usada para orientação e não refletem as limitações físicas do corpo rígido (PAUL, 1981). Uma singularidade matemática ocorre no limite das duas soluções. Nesta orientação singular α e β descrevem a mesma rotação e não podem ser calculados separadamente. Quando $\beta = 0\,^\circ$ ou $180\,^\circ$, taxas infinitas de rotação (α, β, γ) são necessárias para produzir uma velocidade angular finita de corpo rígido.

A derivada temporal do vetor de ângulo de Euler é o vetor de taxas angulares de Euler. A relação entre as taxas angulares de Euler e a velocidade angular do corpo é codificada na matriz de taxas angulares de Euler. Multiplicar essa matriz pelo vetor de taxas angulares de Euler fornece a velocidade angular nas coordenadas globais (DIEBEL, 2006).

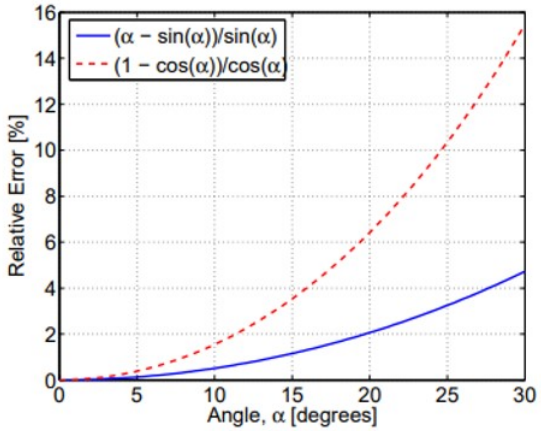

Figura 6 – Erro nas aproximações linearizadas do seno e do cosseno em função do ângulo de entrada (DIEBEL, 2006).

2.9 Quaternion

Um Quaternion é um sistema de quatro parâmetros para especificar a dinâmica de um corpo rígido em relação a algum referencial. Os quatro parâmetros propriamente ditos são determinados resolvendo as equações de taxa de quaternário, que são equações diferenciais lineares cujos coeficientes são as taxas angulares do corpo. Uma vez determinados os parâmetros, os elementos da matriz cosseno que determinam a direção do corpo podem ser formulados. A matriz de cosseno de direção em termos dos parâmetros de quaternion e as equações de taxa de quaternion são ambas derivadas de um modo físico simples usando o conceito do eixo e ângulo de Euler. Representações de quatro parâmetros estão relacionadas aos elementos da matriz de rotação e existem transformações inversas para computar esses parâmetros a partir de uma dada matriz de rotação (GRUBIN, 1970).

2.10 Pitch, Roll e Yaw

Os ângulos de Euler (α, β, γ)e os ângulos roll-pitch-yaw (inclinação) são usados rotineiramente para representar a orientação de corpos rígidos na indústria aeroespacial, navegação e robótica porque minimizam a dimensionalidade do problema de controle (HEMAMI, 1982). O MPU-6050 possui um giroscópio que é usado para medir ângulos de rotação de 3 eixos: (1) ângulo de rotação do eixo X chamado roll , (2) ângulo de rotação do eixo Y chamado pitch e (3) ângulo de rotação do eixo Z chamado yaw (IBRAHIM; ALY; FAR, 2016).

Um corpo rígido possui três graus de liberdade de rotação, três parâmetros independentes devem ser suficientes para caracterizar completa e inequivocamente sua orientação.Representações de três parâmetros são populares em engenharia porque minimizam

a dimensionalidade do problema de controle de corpo rígido. As representações de orientação mais comumente usadas são os ângulos de Euler e os ângulos de inclinação do rolo.Os ângulos de Euler e de pitch-roll-yaw da inclinação são relacionados aos elementos da representação da matriz de rotação e uma transformação inversa é usada rotineiramente para calcular esses ângulos de orientação a partir de uma determinada matriz de rotação (ANG; TOURASSIS, 1987).

O giroscópio fornece medições precisas que não estão sujeitas ou são suscetíveis a forças externas. No entanto, quando o sistema retorna à sua posição original, há uma tendência a desviar e não retornar a zero. Isto é devido à integração ao longo do tempo (IBRAHIM; ALY; FAR, 2016).

Em Jefiza et al. (2017), é possível notar as seguintes equações de inclinação, para se calcular os valores de pitch, roll e yaw :

$$Roll = \arctan\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right) * \frac{180}{\pi} \tag{2.6}$$

$$Pitch = \arctan\left(\frac{a_y}{\sqrt{a_x^2 + a_z^2}}\right) * \frac{180}{\pi} \tag{2.7}$$

$$Yaw = \arctan\left(\frac{\sqrt{a_x^2 + a_z^2}}{a_z}\right) * \frac{180}{\pi} \tag{2.8}$$

2.11 Python

Python é uma linguagem originalmente projetada por Guido van Rossum para facilitar o aprendizado; Van Rossum sugeriu que todos poderiam dominar a programação usando Python (ROSSUM et al., 1999).Possui muitos dos recursos característicos de uma linguagem acessível e adequada para programação, por exemplo, dispõe de uma sintaxe breve e clara, comparando-se a como Java ou C ++, tem uma sintaxe mais intuitiva; sua capacidade de digitação dinâmica, reduz ainda mais a notação; apresenta feedback imediato e de fácil compreensão sobre possíveis erros exemplificando rapidamente conceitos de programação; impõe uma maneira indentada e estruturada de escrever códigos; gratuito, open source com uma grande quantidade de tutoriais, livros, exercícios, tarefas e extensa documentação disponível na web (GRANDELL et al., 2006).

Durante a última década, o Python (uma linguagem de programação interpretada e de alto nível) tornou-se, sem dúvida, o padrão de fato para pesquisas científicas exploratórias, interativas e baseadas em computação. Embora o Python não tenha sido projetado especificamente para atender às necessidades computacionais da comunidade científica, ele rapidamente atraiu o interesse de cientistas e engenheiros. Possui uma sintaxe expressiva

e uma rica coleção de tipos de dados integrados (como strings),e detém uma variedade extensa de bibliotecas de propósito geral (MILLMAN; AIVAZIS, 2011).

A sintaxe clara do Python é capaz de tornar o código fácil de compreender e estruturar. Algumas das singularidades dessa sintaxe incluem blocos de código definidos por recuo, amplo uso de namespaces(módulos) e construções de loop (laços) fáceis de ler. Os módulos são arquivos (com um sufixo ".py") com código Python ou bibliotecas compartilhadas compiladas criadas especificamente para importação em Python, .Cada módulo contém um namespace que permite o agrupamento de funções, classes e variáveis semelhantes de uma maneira que permite que o código seja dimensionado de scripts simples para programas complexos.Uma sequência adicional em Python é uma lista mutável (pode ser alterada).Assim, podemos usar listas para construir matrizes multidimensionais simples.Essa é uma maneira rápida e eficiente de armazenar e trabalhar com pequenas matrizes.Também contribui para a construção de um código de fácil manutenção, separando em grupos lógicos, como módulos, classes (novas definições de objeto) e funções. Suporta diversos estilos de programação, tanto procedurais quanto orientados a objetos.Em cada módulo, você pode definir funções para implementar o comportamento ou criar novos objetos definindo classes. Esses novos objetos podem ter métodos e atributos, incluindo métodos especiais que ensinam ao Python como interpretar a sintaxe específica do objeto. As classes contêm atributos acessados via notação de ponto (object.attribute), e os métodos são atributos que podem ser chamados como uma função. Eles são definidos dentro de um bloco de classes para tomar o próprio objeto como o primeiro argumento. Este objeto não é necessário ao chamar o método em uma instância da classe, portanto, para chamar o método tolist () do objeto de uma classe vetorial (que é ligado ao nome "v"), você escreveria v.tolist () (OLIPHANT, 2007).

2.12 MultiArray

Matrizes multidimensionais são frequentemente usadas, mas a linguagem nativa fornece suporte limitado para elas. Em sua Biblioteca Padrão, fornece interfaces sofisticadas para manipular dados sequenciais, mas depende de sua herança C básica para matrizes. A biblioteca MultiArray, uma parte da coleção de bibliotecas adicionais, aprimora o conjunto de ferramentas de um programador com abstrações de matriz multidimensionais versáteis. Ele inclui um modelo de classe de matriz geral e adaptadores de matriz nativa que oferecem suporte a operações de matriz idiomática e inter-operam com os algoritmos da Biblioteca Padrão . As matrizes compartilham uma interface comum, expressa como um conceito de programação genérica, em termos de quais algoritmos genéricos de matrizes podem ser implementados (GARCIA; LUMSDAINE, 2005).

Matrizes são um recurso universal de linguagens de programação de propósito ge-

ral. Desempenham um papel proeminente nas linguagens modernas em uso atualmente, como Java, Python e C ++. A biblioteca padrão não fornece nenhum componente similarmente benéfico para matrizes multidimensionais. A biblioteca MultiArray cumpre essa função. Ele implementa uma forma de matriz multidimensional de propósito geral e componentes para adaptar matrizes nativas a uma interface semelhante. A biblioteca também estabelece uma base para programação genérica com matrizes, define o conceito MultiArray, uma especificação para interfaces de matriz.Uma matriz pode ser visualizada como uma hierarquia aninhada de contêineres. Sua profundidade de aninhamento determina seu número de dimensões ou dimensionalidade.As matrizes contidas em outro matriz, são chamadas de suas submatrizes ou sub-arranjos (subarray). Uma matriz bidimensional tem sub-arranjos unidimensionais. O tipo de valor de uma matriz ou qualquer outro contêiner é o tipo de seus valores contidos imediatamente. O tipo de elemento de uma matriz é o tipo dos objetos em seu aninhamento mais interno (SOLDOVIERI; SOLIMENE; PRISCO, 2008).

Além da funcionalidade de indexação esperada das matrizes, o MultiArray adiciona suporte para visualizações e submatrizes por meio do qual uma parte de uma matriz existente pode ser manipulada como uma matriz de primeira classe. O conceito MultiArray formaliza uma interface genérica para matrizes contra as quais algoritmos genéricos que podem operar em qualquer tipo de arranjo MultiArray podem ser implementados. Apresenta uma interface amigável, eficaz e versátil para o desenvolvedor. Como todas as matrizes nessa biblioteca modelam o conceito, suas expressões válidas e tipos associados podem ser usados para implementar algoritmos genéricos que funcionam com qualquer componente MultiArray. Modelos de Multiarray usam colchetes para endereçar sub-arranjos.A classe multi array é definida no arquivo de cabeçalho da biblioteca, que fornece todos os componentes necessários para usar várias matrizes (GARCIA; LUMSDAINE, 2005).

2.13 Numpy

Em primeiro lugar, o NumPy fornece uma matriz homogênea e multidimensional de um tipo de dados específico.Embora o principal objetivo da matriz seja manter itens do mesmo tipo, um dos tipos de dados internos disponíveis é chamado de "objeto". Uma matriz dele pode conter um objeto Python arbitrário em cada elemento, portanto, é efetivamente um matriz multidimensional heterogênea. Os tipos de dados que uma matriz pode conter é bastante arbitrário. A matriz do NumPy suporta internamente todos os tipos de dados fundamentais do C, incluindo 10 tipos diferentes de inteiros, três tipos de floats, três tipos de números complexos, e um tipo booleano. Além disso, as matrizes podem conter strings ou strings unicode, e você pode até mesmo definir seu próprio tipo de dados que é equivalente a uma estrutura C (às vezes chamada de registro). Dada a complexidade potencial do tipo de dados da matriz, é útil pensar em uma matriz no

NumPy como uma coleção de itens que consuma exatamente o mesmo número de bytes. Um objeto de tipo de dados descreve cada elemento na matriz, enquanto a própria matriz fornece as informações sobre sua forma.Para matrizes grandes, o NumPy é muito adequado para gerenciar requisitos de memória e velocidade (OLIPHANT, 2007).

Em Walt, Colbert e Varoquaux (2011) afirma-se que as matrizes NumPy podem ter até 32 dimensões; Também podem conter outros tipos de elementos (ou até mesmo combinações de elementos), como booleanos ou datas. É uma maneira conveniente de descrever um ou mais blocos de memória do computador, de modo que os números representados possam ser facilmente manipulados. O NumPy fornece uma abstração de alto nível para computação numérica sem comprometer o desempenho.É importado da seguinte forma:

import numpy as np

Em Python, é possível indexar os elementos de uma matriz usando o operador equivalente a [ ]. Os tipos de elementos contidos na matriz, podem ser float ou inteiros. O modelo de memória distribuída do NumPy, fornece uma maneira poderosa de visualizar a mesma memória de maneiras diferentes sem copiar dados. Em seguida, há um exemplo de um modelo de abordagem vetorizada aplicado para todos os elementos de uma matriz, onde In[ ] são entradas e Out[ ] saída:

In [1]: a = np.array([1, 3, 5])

In [2]: b = 3 * a

In [3]: b

Out[4]: array([3, 9, 15])

Operações vetorizadas no NumPy são implementadas em C e Python, resultando em uma melhoria significativa de velocidade. Essas operações não estão restritas a interações entre escalares e matrizes. Por exemplo, o NumPy pode executar uma subtração rápida de elementos de dois arrays da seguinte forma:

In [5]: b – a Out[6]: array([2, 6, 10])

2.14 ROS

"O ROS é um sistema meta-operacional de código aberto para o seu robô. Ele fornece os serviços que você esperaria de um sistema operacional, incluindo abstração de hardware, controle de dispositivo de baixo nível, implementação de funcionalidade comumente usada, passagem de mensagens entre processos e gerenciamento de pacotes. Também fornece ferramentas e bibliotecas para obter, criar, gravar e executar código em vários computadores" (INDUSTRIAL, 2009).É um sistema operacional completo para

robótica de serviços.O ROS é, na verdade, algo entre um sistema operacional e um middleware.Fornece não apenas serviços padrão do sistema operacional (abstração de hardware, gerenciamento de contenção, gerenciamento de processos), mas também funcionalidades de alto nível (chamadas assíncronas e síncronas, banco de dados centralizado, um sistema de configuração de robôs) (MAZZARI, 2016).

O Robot Operating System (ROS) em outras palavras, é um framework amplamente utilizado no mundo da robótica. Sua modularidade oferece aos usuários a capacidade de reutilizar o código em diferentes plataformas sem a necessidade de grandes mudanças, permitindo a transferência de funcionalidades entre diferentes robôs (PEREZ, 2017).

Desde o início, o desenvolvimento de ROS não foi centralizado, uma vez que foi desenvolvido por diferentes instituições. Cada desenvolvedor pode desenvolver novas bibliotecas para o ROS e compartilhá-las com os outros membros da comunidade, sem a necessidade de permissões de publicação ou de qualquer controle por parte da comunidade. Isso permite um desenvolvimento muito mais dinâmico, a capacidade de receber feedback e melhorar as bibliotecas fornecidas(PEREZ, 2017).

Módulos de software ROS pode ser escrito em qualquer idioma para o qual uma biblioteca cliente foi escrita. Existem bibliotecas cliente para C ++, Python, Java, JavaScript, Ruby, LISP e MATLAB, entre outros. As bibliotecas do ROS podem se comunicar entre si seguindo uma convenção que descreve como as mensagens são "achatadas"antes de serem transmitidos.Este sistema operacional consistem em vários pequenos programas de computador que conectam-se entre si e trocam mensagens continuamente. Estas mensagens viajam diretamente de um programa para outro; não há serviço de roteamento central. Embora isso faça o "encanamento"subjacente mais complexo, o resultado é um sistema que escala melhor à medida que a quantidade de dados aumenta(QUIGLEY; SMART, 2010).

Um dos pontos fortes do ROS é o poderoso conjunto de ferramentas de desenvolvimento. Várias ferramentas estão incluídas que fornecem introspecção, depuração, plotagem e visualização de variáveis. As duas ferramentas mais conhecidas da ROS são rviz (para experimento de visualização) e rqt (para visualização de dados e incorporação de módulos gráficos).Fornecem capacidades de depuração adicional. Finalmente, o ROS fornece uma integração perfeita com outras bibliotecas robóticas aceitas pela comunidade, como o simulador Gazebo 3D, OpenCV para imagem processamento, PCL (biblioteca de nuvem de pontos) e MoveIt! para navegação. Deve-se afirmar que o ROS não é em tempo real, mas o código em tempo real pode ser incorporado. Além disso, em 2009, a integração do ROS e Orocos RTT (estrutura em tempo real) foi anunciado (TSARDOULIAS E. ; MITKAS, 1998).

Os componentes fundamentais em aplicações baseadas em ROS são nós que se

comunicam através de um paradigma publicador-receptor, onde as mensagens são organizados em tópicos nomeados. Um nó (por exemplo, um sensor) compartilha informações publicando mensagens sobre o tópico apropriado, enquanto nós que desejam receber informações (por exemplo, um atuador) inscrever-se nos tópicos relevantes.O ROS certifica-se que os publicadores e os receptores se conectem para estabelecer comunicações par a par (HALDER et al., 2017).

O sistema operacional ROS suporta um grande número de diferentes sensores, marcas e funcionalidades, e todos eles possuem uma fração específica no sistema, e caso o sensor que você necessite usar não seja suportado, há a opção de gerar sua documentação exigida para o uso. Por essa facilidade de acomodação de vários dispositivos e versatilidade que a ROS possui, que escolhemos este sistema operacional.

2.15 ROS na reabilitação

A plataforma ROS apresenta um futuro promissor no campo da reabilitação. Devido às suas vantagens, que são trazidas ao processo de formação, facilitam os procedimentos para criação de softwares de assistência ao próximo e melhoria da qualidade de vida. A incorporação de sistemas robóticos em um ambiente não industrial, como hospitais e centros de reabilitação, tem que tornar-se cada vez mais difundida. A robótica de reabilitação, aborda o estudo de sistemas robóticos complexos com o objetivo de restaurar as funções para aquelas pessoas que sofrem grandes traumas como resultado de acidentes vasculares cerebrais e doenças cerebrovasculares. Na reabilitação é uma abordagem valiosa para apoiar os fisioterapeutas, intensificar a terapia e possibilitar a independência, reprodutibilidade e treinamento(J.RODRIGUEZ-LERA LEANDRO GOMES, 2018).

Manter o interesse de pacientes que necessitam de programas longos de reabilitação requer muita atenção e cuidado, pois a partir de certo momento, a repetição dos exercícios se torna cansativa e monótona, e a frequência tende a cair, principalmente em pacientes que já apresentam alguma melhora, e acreditam que já seja suficiente o tratamento naquele patamar, parando de comparecer às sessões. Assim, o uso de robôs com a finalidade de melhorar a motivação do paciente durante os exercícios de reabilitação são de extrema necessidade e eficiência(ANDRIST BILGE MUTLU, 2015).

A integração da realidade virtual ao ambiente hospitalar, sem dúvida, traz mais diversão para os pacientes e os torna mais envolvidos durante repetidos movimentos de treinamento. É indicado e orientado especialmente para tarefas que incluam exercícios dentro da realidade virtual, utilizado juntamente a reabilitação robótica, conseguindo assim, fornecer uma experiência terapêutica mais divertida que pode ativar o desejo de movimento ágil dos pacientes e promovê-los a colocar seu próprio esforço, em auxílio durante o terapia(LAVER, 2011).

Esses métodos de reabilitação não são tão abertos para personalizar convenientemente o ambiente virtual.A partir daqui são outros exempos de como o ROS é aplicado na robótica de reabilitação, é possível utilizar a ROS, por exemplo, para desenvolvimento de um ambiente de treinamento virtual para reabilitação de membros. Desta forma, com a mobilidade e abordagem multilíngue oferecida pelo sistema, por ser uma plataforma open source, é aberta o suficiente para ser modificado com incomplexidade e, vários modelos podem ser adicionados ou removidos assim que for apropriado.

A tecnologia robótica deve ser aplicada visando ampliar a quantidade de pessoas com doenças sérias que conseguiram vencê-la ou adiquiriram mais qualidade de vida. É possível citar o acidente vascular cerebral, como uma das doenças mais freqüentes entre os idosos e muitas vezes leva a uma falha contínua de funções no sistema nervoso central. Devido à plasticidade do cérebro afetados podem recuperar a função motora perdida por treinamento repetitivo.Os dispositivos robóticos podem ser uma abordagem para acelerar o processo de reabilitação, maximizando a intensidade de treinamento dos pacientes(ESCHERT; DISSELHORST-KLUG, 2018).

Na reabilitação de pacientes que foram diagnosticados com uma doença cerebrovascular (CVA). Um conjunto de disfunções cerebrais relacionadas com os vasos sanguíneos e o fluxo de sangue no cérebro, causando perda total ou perda de amplitude de movimento, diminuição no tempo de reação e organização do movimento desordenado , déficits no controle motor, que afetam diretamente a independência do paciente. Em Popescu Vlad Ciobanu (2016) uma luva robótica háptica, foi desenvolvida, visando melhorar a recuperação, com programas que utilizam um conjunto de exercícios de reabilitação pré-definidos. Eles trabalharão juntos e, de acordo com o movimento da mão, e o progresso será alcançado quando o paciente usar mais sua própria força e menos a luva controlada.Com a integração da luva robótica háptica,o software Shadow Dexterous Hand, e ROS (Robot Operating System) ,os movimentos feitos por um paciente portando a luva, serão encontrados no ambiente do simulador de realidade virtual.

Há também aplicações em robôs socialmente assistivos que são planejados para fornecer serviços sociais e assistência cognitiva onde eles buscarão motivar e engajar pessoas em atividades terapêuticas. Em Andrist Bilge Mutlu (2015) foi usado o sistema operacional (ROS) para lidar com a execução e comunicação entre cada um dos componentes do sistema. Foi desenvolvido um robô com comportamento projetado analisando o olhar de seres humanos reais, com conduta que permite ao robô combinar a dimensão extroversão da personalidade com o paciente. Assim, mais efetivamente motivando os desenvolvedores a se engajarem repetidamente em uma tarefa terapêutica.

2.16 Contextualização do desenvolvimento inicial do protótipo

A proposta de um sistema modular de medição do movimento humano baseado em ROS e sensores inerciais visa colaborar no campo da biomecânica humana voltado para o estudo da cinesiologia, que segundo Burke (1977), pode ser definida como o estudo do comportamento do movimento de todos os organismos vivos.

Lianza (2007) cita, que os objetivos da reabilitação ortopédica são a analgesia, a manutenção e o ganho da amplitude de movimento e da força muscular, a profilaxia e o tratamento das deformidades articulares, visando a recuperação funcional e a independência para realização das atividades da vida prática e diária . Tendo em vista a grande demanda de cirurgias ortopédicas que geralmente tratam de reparos ou implantes nas articulações, nas quais, a reabilitação do paciente depende diretamente da recuperação dos movimentos temporariamente perdidos, esse projeto pretende desenvolver um sistema portátil de avaliação do movimento humano, utilizando no hardware, duas Unidades de Medição Inercial (IMU), GY-521, e uma raspberry Pi 3.

Os dados obtidos pelos sensores inerciais podem representar uma vantajosa fonte de informações. Clinicamente relevantes, são capazes de fornecer detalhes para uma avaliação motora nos indivíduos que realizam processo de reabilitação, por exemplo. Em ambientes clínicos, a avaliação é geralmente conduzida através de medidas como uma avaliação visual ou um questionário (O'FLYNN et al., 2018). Utilizando sensores inerciais é possível fornecer uma medida objetiva e empírica do progresso de um paciente através da reabilitação, com sensores vestíveis para o monitoramento do movimento.

Esses sensores inerciais exibem uma notória vantagem em pontos como, dimensão reduzida, peso e custo. Porém as estimativas fornecidas podem ser corrompidas por acúmulo de pequenos erros (BO; HAYASHIBE; POIGNET, 2011).

Em Quigley et al. (2009), O ROS - Robot Operating System , foi projetado para ser compatível com diversos sistemas operacionais de acordo com Pinto et al. (2015). Selecionada para unificar os processos, a plataforma foi essencial para estruturação do protótipo. Sucintamente, a comunicação do sistema é estabelecida via "nós" (node). Se comunicam uns com os outros, passando mensagens, que são caracteres em formato padrão como, float, int, boolean. As mensagens são enviadas via nós e publicadas para um determinado tópico (topic). De acordo os dados recebidos no nó, ele se relacionará diretamente com o tópico apropriado. O nó pode publicar/receber vários tópicos (QUIGLEY et al., 2009).

3 Metodologia

3.1 Introdução ao desenvolvimento do protótipo

Esse trabalho propõe o desenvolvimento de um sistema para medição dos ângulos da articulação do cotovelo humano utilizando sensores incerciais e baseado em um sistema operacional robótico (ROS), representado na figura 7.

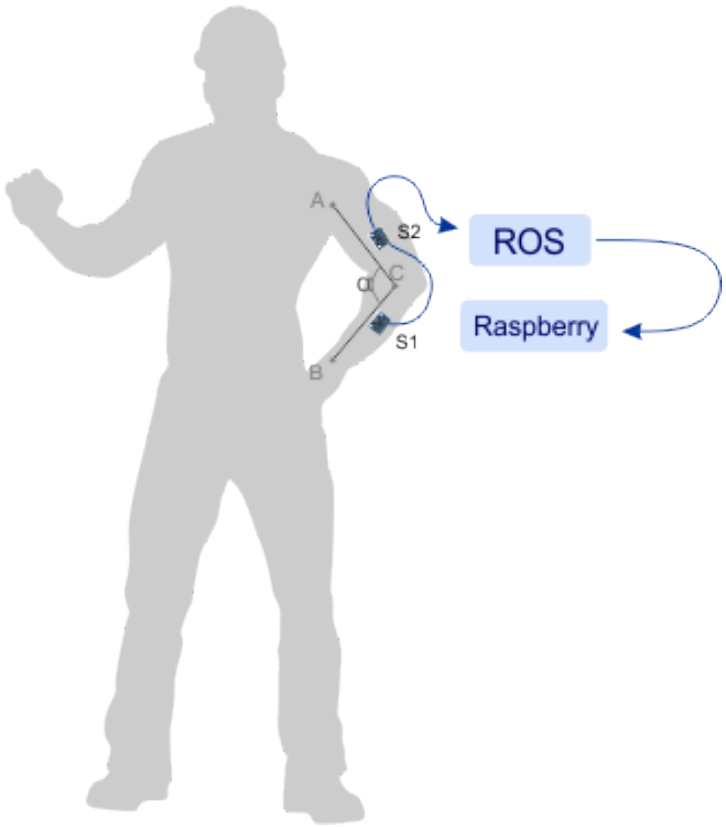

Figura 7 – Localização dos sensores no braço e ângulo a ser medido. (Autoria própria).

Para orientação e medição foi adquirido dois módulos GY-521 (S1 e S2). A articulação que será medida é aquela representada pelo ângulo α. Compondo o sistema de integração das IMUs o ROS fará toda a interconexção possibilitando o acoplamento de vários sensores e, também, o processamento de diversos códigos em linguagens diferentes que poderá ser usado futuramente. O hardware escolhido foi a Raspberry Pi que apresentou preço acessível e compatibilidade com os requisitos do projeto. Nesta etapa inicial serão aferidos os ângulos da articulação do cotovelo, e em trabalhos posteriores é planejado estender à outras articulações do corpo. Por meio dos dados coletados das IMUs o sistema proposto irá realizar a medição em tempo real da posição/orientação relativa da articulação mencionada anteriormente, mostrando o posicionamento adequado

do movimento do membro

3.2 Software

É necessário, para realizar a conexão entre as várias IMUs do sistema modular, um sistema de integração. Para este trabalho, este sistema será a criação de nós na ROS que fará a interligação dos sensores inerciais junto com a aquisição dos ângulos. Como o sistema fará o uso de vários sensores inerciais, primeiramente utilizando duas IMUs, será necessário a replicação dos nós para acomodar múltiplas unidades de medição inercial.

Inicialmente, assimilou-se os conceitos dos tutoriais encontrados em ROS Nodes () e ROS msg and srv (),respectivamente "Understanding ROS Nodes" e "Creating a ROS msg and srv". Seguiu-se então, o tutorial "Writing a Simple Publisher and Subscriber (Python)". Em OSRF (), exemplifica-se a estruturação de um código "publisher and subscriber", desde os comandos para a criação de pastas para armazenamento até o modelo padrão de arranjo a ser compilado. Assim, após relacionar as informações adquiridas, foi possível compreender o funcionamento e realizar a estruturação de uma publicação de uma string em um nó, e ao mesmo tempo em outro nó foi feita a leitura dessa string. Em MrTijn/Tijndagamer () a biblioteca do sensor MPU-6050 pode ser observada e, a mesma foi adaptada no código disposto da ROS para este sensor especificamente, dentro de um ambiente catkin. O primeiro teste realizado foi captando os valores puros enviados de um módulo. Criou-se então o nó 'tcc1' para o primeiro MPU-6050 que foi testado. Através da linha:

$$rospy.init\_node('tcc1', anonymous = False)$$

Para testar o código .py, deve-se, iniciar o roscore executando as seguintes linhas de comando no terminal MATE :

$source \quad /opt/ros/kinetic/setup.bash$

$cd \quad catkin\_ws/$

$source \quad devel/setup.bash$

$echo \quad \$ROS\_PACKAGE\_PATH$

$roscore$

A pasta beginner_tutorials foi criada para armazenar os códigos desenvolvidos. Na segunda aba do terminal, é executado o programa "sensor1.py" elaborado:

$cd \quad catkin\_ws$

$cd \quad src$

$cd \quad beginner\_tutorials$

cd scripts

source /opt/ros/kinetic/setup.bash

python sensor1.py

A fim de conferir se a publicação está sendo feita dentro da ROS, é primordial abrir um terceiro terminal no qual o código estará funcionando dentro da mesma. Assim, é necessário reproduzir alguns comandos, onde é chamado o tópico criado, no caso do primeiro MPU-6050, imu_upper:

rostopic list

rostopic echo /imu_upper

Posteriormente, construiu-se um sistema capaz de compreender os dados puros e publicá-los em Ângulos de Euler dentro da ROS. O passo seguinte consistiu em desenvolver um publisher e um listener capazes de publicar e receber os dados publicados dos dois IMU's e relacioná-los publicando os valores de pitch e roll. Porém, antes é importante citar as adaptações realizadas para o programa do segundo MPU-6050 conectado. Para o código do sensor2.py , o endereçamento do sensor é o 0x69, que é definido na linha:

Device_Address = 0x69

Na linha abaixo, um novo nó intitulado tcc2 é criado :

rospy.init_node('tcc2', anonymous=False)

Da mesma maneira que é necessário iniciar um novo tópico, denominado imu_lower, a partir da mesma linha de comando que define a classe do objeto publicado como Float64MultiArray:

pub = rospy.Publisher('/imu_lower', Float64MultiArray, queue_size=10)

No começo do código, após a importação de bibliotecas e termos de extrema significância ao trabalho, o endereço de alguns registradores é definido seguindo a biblioteca própria do sensor em questão. Como pode ser visto a seguir:

PWR_MGMT_1 = 0x6B

SMPLRT_DIV = 0x19

CONFIG = 0x1A

GYRO_CONFIG = 0x1B

INT_ENABLE = 0x38

ACCEL_XOUT_H = 0x3B

ACCEL_YOUT_H = 0x3D

ACCEL_ZOUT_H = 0x3F

GYRO_XOUT_H = 0x43

GYRO_YOUT_H = 0x45

GYRO_ZOUT_H = 0x47

Após as delimitações realizadas, os registradores citados abaixo foram pré-definidos. Configurou-se posições fixas via dados recebidos:

```
def MPU_Init():
#Grava no registrador de taxa de amostragem
bus.write_byte_data(Device_Address, SMPLRT_DIV, 7)
#Grava no registrador de gerenciamento de energia
bus.write_byte_data(Device_Address, PWR_MGMT_1, 1)
#Grava no registrador de configuração
bus.write_byte_data(Device_Address, CONFIG, 0)
#Grava no registrador de configuração do Giroscópio
bus.write_byte_data(Device_Address, GYRO_CONFIG, 24)
#Grava no registrador que interrompe uma habilitação
bus.write_byte_data(Device_Address, INT_ENABLE, 1)
```

No bloco seguinte, os dados são fornecidos do sensor como valores curtos de 16 bits (-32768 a +32767) divididos em dois registradores de 8 bits (0 a 255). Esta parte do código está lendo o par de registradores de 8 bits e convertendo os resultados em um valor de 16 bits. Existem 6 pares de registradores como este para o X, Y, Z do acelerômetro e giroscópio.

```
def read_raw_data(addr):
    high = bus.read_byte_data(Device_Address, addr)
    low = bus.read_byte_data(Device_Address, addr+1)
    value = ((high << 8)  |  low)
```

Posteriormente, em vez de ler os dois bytes separadamente, ele está lendo-os como um valor de 16 bits não assinado (0 a 65535) e convertendo-o em um valor assinado (-32768 a +32767). A leitura dos dados puros é efetuada e no final esse valor retorna.

```
if(value > 32768):
  value = value - 65536
return value
```

Prosseguindo para o bloco de configuração da estrutura de publicação. Onde define-

se as variáveis correspondentes aos registradores que realizarão as respectivas leituras dos valores puros.

#Leitura dos valores puros do acelerômetro

acc_x = read_raw_data(ACCEL_XOUT_H)

acc_y = read_raw_data(ACCEL_YOUT_H)

acc_z = read_raw_data(ACCEL_ZOUT_H)

#Leitura dos valores puros do Giroscópio

gyro_x = read_raw_data(GYRO_XOUT_H)

gyro_y = read_raw_data(GYRO_YOUT_H)

gyro_z = read_raw_data(GYRO_ZOUT_H)

Seguindo as demarcações encontradas na biblioteca [54], elaborou-se com base nas constantes definidas por ela, os seguintes cálculos:

Ax = acc_x/16384.0

Ay = acc_y/16384.0

Az = acc_z/16384.0

Gx = gyro_x/131.0

Gy = gyro_y/131.0

Gz = gyro_z/131.0

Os valores de Ax, Ay, Az, Gx, Gy, Gz são os valores reais recebidos do acelerômetro e giroscópio após serem dimensionados. Em seguida, os valores de Gx e Gy são usados para calcular o ângulo de inclinação (pitch e roll) percorrido e adicionam esse dado às variáveis angle_pitch e angle_roll. O valor 0.0000611 previamente definido equivale a 1 / (250Hz * 65.5), valores referentes às definições dimensionais configuradas no código e estipuladas na biblioteca, também podem ser conferidas em Arki et al. (2017).

*angle_pitch = Gx * 0.0000611*

*angle_roll = Gy * 0.0000611*

Se a IMU realizar o movimento inclinação yaw, que é definido por um movimento de rotação ao redor do eixo z de um corpo rígido, o ângulo roll deve ser transferido para o ângulo pitch, e vice-versa. O valor de 0.000001066 que é multiplicado pelos dados do giroscópio, corresponde ao cálculo (0.0000611 * (π / 180) (ARKI et al., 2017).

*angle_pitch = angle_roll * math.sin(Gz * 0.000001066)*

*angle_roll = angle_pitch * math.sin(Gz * 0.000001066)*

Os cálculos angulares do acelerômetro são executados a seguir. Ressaltando o valor constante 57.296 que consiste na solução de 1 / (π / 180). A função np.arcsin corrigi o desvio dos ângulos de inclinação pitch e roll do giroscópio com o respectivo ângulo de inclinação e rotação do acelerômetro.

*acc_total_vector = math.sqrt((Ax * Ax) + (Ay * Ay) + (Az * Az))* #Cálculo do vetor total do acelerômetro

*angle_pitch_acc = np.arcsin(Ay / acc_total_vector) * 57.296* #Cálculo do ângulo pitch

*angle_roll_acc = np.arcsin(Ax / acc_total_vector) * -57.296* #Cálculo do ângulo roll

ângulo roll

A calibração automática é configurada seguindo as diretrizes observadas em Laboratório de Robótica (LARA) (), o valor da calibragem do acelerômetro para pitch e roll é verificado nas linhas abaixo:

angle_pitchz_acc -= 1.1

angle_roll_acc -= -2.73

Os dados são enviados um por vez, primeiro os dados de pitch via msg1 = angle_pitch_acc e em seguida os de roll , msg2 = angle_roll_acc. A função msg recebe os dados e publica como um caracter Float64MultiArray.

Construiu-se um listener que recebe os dados enviados dos dois MPU6050, identificando-os e realizando a subtração de vetores necessária que é o valor final a ser publicado. Segundo os atributos de classe definidos em Python e seguindo o modelo de subscriber já citado, estruturou-se um programa capaz de combinar os dados recebidos e publicá-los após os cálculos primordiais serem realizados. Inicialmente, definiu-se os valores que seriam recebidos de pitch como self.upper e roll como self.lower, a contagem foi delimitada para começar em 0.

```
def ___init___(self):
    self.upper = 0
    self.lower = 0
```

No bloco seguinte, relaciona-se os dados recebidos de cada MPU6050 com os nomes das constantes intituladas no bloco anterior.

```
def upper_callback(self,msg1):
    self.upper0 = msg1.data[0]
    self.upper1 = msg1.data[1]
```

```
def lower_callback(self,msg2):
    self.lower0 = msg2.data[0]
    self.lower1 = msg2.data[1]
```

A diferença do primeiro sensor é calculada em dif0 e do segundo em dif1.

```
dif0 = self.upper0 - self.lower0
dif1 = self.upper1 - self.lower1
```

Os dados das duas diferenças são recebidos pela função dif que realiza a diferença total e publica através da linha descrita a seguir:

```
dif = [dif0, dif1]
rospy.loginfo(dif)
```

No próximo bloco foi definido as características específicas do listener, onde um novo nó foi iniciado e os dados são lidos dos tópicos imu_upper e imu_lower, no formato Float64MultiArray , chamando individualmente os respectivos dados dos blocos de callback (d.upper_callback e d.lower_callback) .

```
def listener(self,d):
    rospy.init_node('listener', anonymous=True)
    rospy.Subscriber("imu_upper", Float64MultiArray, d.upper_callback)
    rospy.Subscriber("imu_lower", Float64MultiArray, d.lower_callback)
    rospy.spin()
```

Desenvolveu-se um código que coleta dados dos sensores inerciais com base na estrutura já citada acima, os valores puros obtidos são transformados e apresentados em Ângulos de Euler. A linguagem Python escolhida, oferece uma sintaxe limpa e intuitiva (ATEEQ et al., 2014).

4 Resultados

4.1 Validação

Para validar os resultados do protótipo proposto foi realizado primeiramente, medições dos ângulos da junta do cotovelo com os sensores acoplados considerando os módulos GY-521 parados com o eixo Z apontado para baixo.

A figura 8 apresenta o protótipo confeccionado para o trabalho proposto. Contendo dois módulos de medições inerciais acoplados ao Rapberry Pi 3.

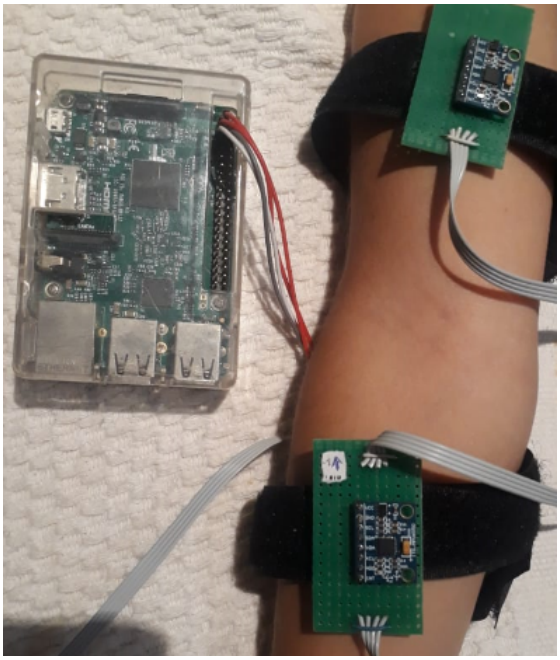

Figura 8 – Protótipo proposto (Autoria própria).

Foram consideradas medições de duas posições do braço. A primeira posição com o braço e antebraço esticados formando aproximadamente um ânglo de 0°.

Figura 9 – Posição 1 do usuário para medição do ângulo da junta do braço pelos sensores (Autoria própria).

A segunda posição com o antebraço flexionado deixando o punho paralelo formando, aproximadamente, um ângulos de 90°. Em cada posição foram consideradas cinco medições dos sensores e posteriormente retirada a média das medições.

Figura 10 – Posição 2 do usuário para medição do ângulo da junta do braço pelos sensores (Autoria própria).

Nas figuras 11 e 12 podemos observar as medições indicadas pelo protótipo das IMUs. Esse protótipo leva em consideração a diferença entre as mediçoes de cada sensor individualmente. No vetor da medição o primeiro elemento é o valor do ângulo Pitch e o segundo do ângulo Roll.

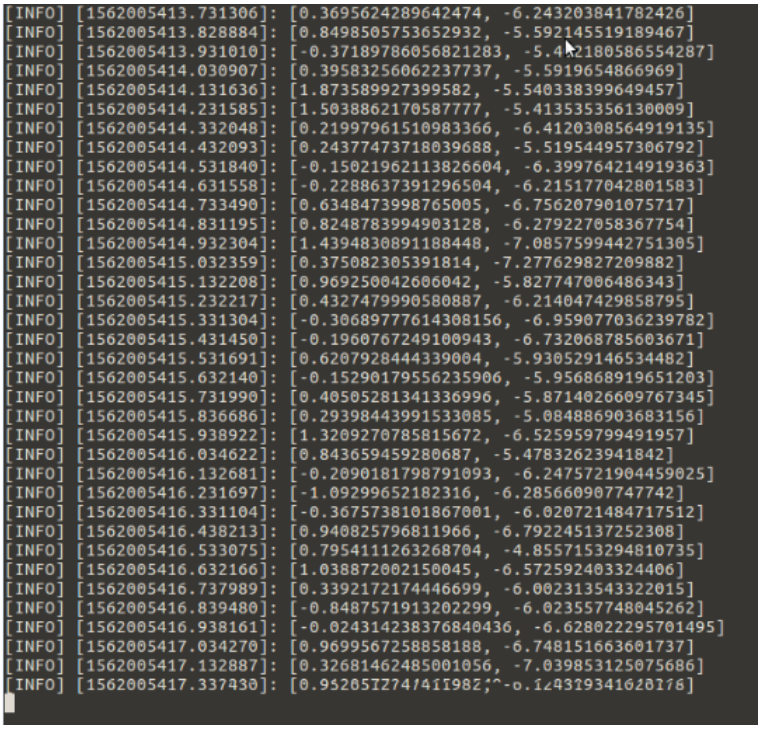

Figura 11 – Resultado da medição na Posição 1 do usuário para medição do ângulo da junta do braço pelos sensores (Autoria própria).

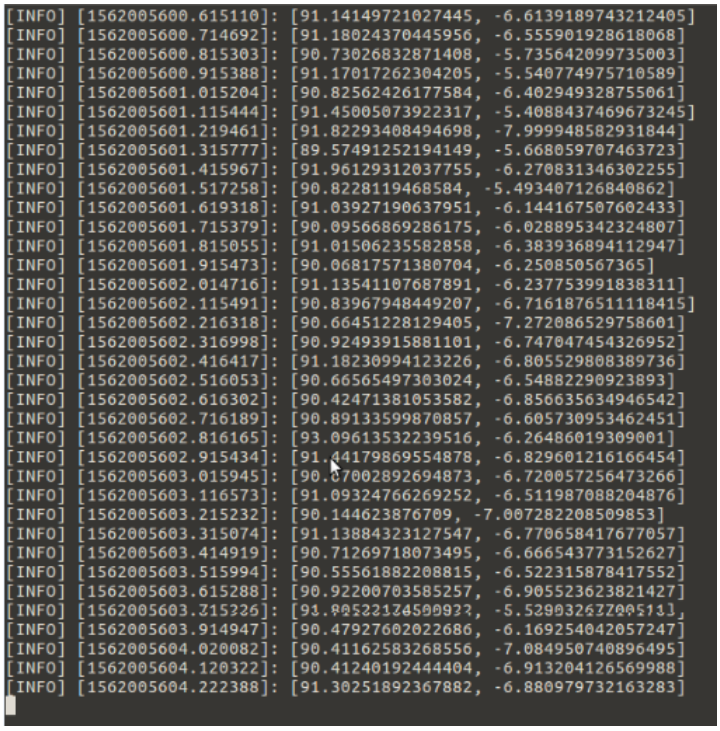

Figura 12 – Resultado da medição na Posição 2 do usuário para medição do ângulo da junta do braço pelos sensores (Autoria própria).

Os resultados obtidos na medição dos sensores foram dispostos em uma tabela, na qual contém, também, a média e desvio padrão das medições.

	Posição 1	**Posição 2**
Medição 1	0,39°	91.17°
Medição 2	1,87°	90.82°
Medição 3	1,50°	91.45°
Medição 4	0,21°	91.82°
Medição 5	0,24°	89.57°
Média	0,84°	90,96°
Desvio Padrão	0,70°	0,77°

Tabela 1 – Medidas referentes as unidades de medição inercial.

Na segunda etapa de validação foi realizado medições com o instrumento goniômetro. Este instrumento é bastante utilizado em análises do movimento humano na área da saúde (FERREIRA, 2010).

Figura 13 – Goniômetro (Autoria própria).

O goniômetro também foi utilizado para medir o ângulo da articulação do cotovelo em duas posições, as mesmas consideradas na medição dos sensores. Nas figuras 14 e 15 podemos observar as posições medidas, bem como a técnica da goniometria.

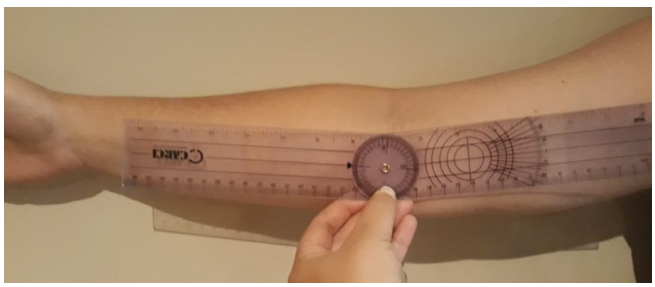

Figura 14 – Medição da Posição 1 utilizando o goniômetro (Autoria própria).

Figura 15 – Medição da Posição 2 utilizando o goniômetro (Autoria própria).

Podemos observar na tabela 2 todas as medições realizadas para a validação do prótipo. Com isso podemos comparar a eficácia do protótipo proposto.

	Medição IMU's		**Medição Goniômetro**	
	Posição 1	**Posição 2**	**Posição 1**	**Posição 2**
Medição 1	0,39°	91.17°	2°	92°
Medição 2	1,87°	90.82°	0°	94°
Medição 3	1,50°	91.45°	1°	91°
Medição 4	0,21°	91.82°	0°	91°
Medição 5	0,24°	89.57°	0°	92°
Média	0,84°	90,96°	0,60°	92°
Desvio Padrão	0,70°	0,77°	0,80°	1,19°

Tabela 2 – Medidas referentes as unidades de medição inercial e goniômetro

Pode-se considerar satisfatórios os resultados obtidos, porém necessitam de melhorias. Uma delas seria a capacidade de medição dos sensores no plano sagital, já que no trabalho em questão os sensores medem apenas no plano horizontal.

A coleta dos dados medidos pelas unidades de medição inercial possui instabilidades relacionadas principalmente a disposição dos sensores no braço e antebraço. Além da sensibilidade do acelerômetro que é proporcional as variações das forças inerciais aferidas por ele. Um fator que também afeta na acurácia das medições é a calibração dos sensores, uma rotina mais eficiênte poderia diminuir os erros relacionados a este fator.

Na utilização do goniômetro para a validação dos dados pode-se considerar, também, fatores que apresentam erros na medição, como o deslocamento do aparelho quando há o movimento das artiulações devido ao tecido mole da pele humana, dificultando o alinhamento correto.

5 Conclusão

Neste trabalho, discorreu-se sobre uma proposta inovadora voltada para um sistema de captura de movimento humano. Apresentou-se a possibilidade de integração do sistema operacional robótico com diversas plataformas, focando-se na finalidade de um produto final móvel, mostrando o comportamento do sistema implementado na placa Raspberry Pi. A vasta quantidade de ferramentas para análise de dados suportada pela plataforma, permite uma maior facilidade de interação entre os sensores e os demais dispositivos de operação do robô. É um sistema muito completo que permite tanto reutilizar códigos antigos, quanto gerar um novo especialmente para o projeto desejado. A eficiência e versatilidade da organização dos comandos internos, e a forma com que a comunicação do sistema é realizada, constituem um pacote absoluto, com características singulares.

Também foi abordado o conceito de IMU e sensores inerciais utilizados para captura do movimento humano. O estudo desses dispositivos possibilitou o conhecimento dessa nova tecnologia, MEMS, bem como a análise de sensores inerciais. Para a medição dos ângulos, esses sensores disponibilizam em um mesmo módulo, giroscópios e acelerômetros que compõem um conjunto de dados que são analisados e processados pelo microcontrolador e a partir disso pode-se estimar a localização de membros do corpo humano. Foi desenvolvido um sistema modular para medição de movimento de membros superiores baseado em ROS e em um conjunto de sensores inerciais.

Conclui-se que o presente trabalho obteve sucesso, apesar das limitações de medição do protótipo e erros relacionados a determinação da posição do ângulo da junta do braço. O protótipo confeccionado obteve êxito no funcionamento e a comunicação I2C dos sensores foi realizada com sucesso. O cálculos dos ângulos utilizando o conceito de ângulos de Euler obteve o resultado esperado com erro relativamente pequeno em comparação ao ângulo medido pelo goniômetro.

Para trabalhos futuros, podemos considerar a melhoria na calibração dos sensores, um protótipo físico mais elaborado, além de uma ferramenta computacional para a análise e apresentação dos dados obtidos pelas IMU's. Com isso, o protótipo poderá ser estentido para além das juntas do braço, podendo medir ângulos de juntas do corpo humano como um todo.

Referências

ANDRIST BILGE MUTLU, A. T. S. Look like me: Matching robot personality via gaze to increase motivation. *CHI 2015, Crossings*, 2015. Citado 2 vezes nas páginas 33 e 34.

ANG, M. H.; TOURASSIS, V. D. Singularities of euler and roll-pitch-yaw representations. *IEEE Transactions on Aerospace and Electronic Systems*, IEEE, n. 3, 1987. Citado 4 vezes nas páginas 8, 24, 26 e 28.

ANUAR, A.; SAHARI, K. S. M.; YUE, E. C. Development of a low cost upper limb motion tracking system with real-time visual output. In: IEEE. *2016 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*. [S.l.], 2016. Citado na página 20.

APOSTOLYUK, V. Theory and design of micromechanical vibratory gyroscopes. *MEMS/NEMS*, 2006. Citado na página 18.

ARDUINO. Mpu6040. *https://www.arduino.cc/*, 2018. Citado 2 vezes nas páginas 8 e 19.

ARKI, A.-B. O. M. A. et al. *Implementation of a Quadcopter Control System*. Tese (Doutorado) — Sudan University of Science and Technology, 2017. Citado na página 40.

ATEEQ, M. et al. C++ or python? which one to begin with: A learner's perspective. In: IEEE. *2014 International Conference on Teaching and Learning in Computing and Engineering*. [S.l.], 2014. Citado na página 42.

BO, A. P. L.; HAYASHIBE, M.; POIGNET, P. Joint angle estimation in rehabilitation with inertial sensors and its integration with kinect. In: IEEE. *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. [S.l.], 2011. Citado 2 vezes nas páginas 14 e 35.

BUONOCUNTO, P.; MARINONI, M. Tracking limbs motion using a wireless network of inertial measurement units. In: IEEE. *Proceedings of the 9th IEEE International Symposium on Industrial Embedded Systems (SIES 2014)*. [S.l.], 2014. Citado na página 14.

BURKE, R. K. The future of kinesiology at the undergraduate level. kinesiology: A national conference on teaching. Urbana-Champaign: University of Illinois, 1977. Citado na página 35.

CALACHE, D. C. Caracterização de um acelerômetro baseado em sistemas microeletromecânicos (mems). *Universidade do Estado do Rio de Janeiro*, 2013. Citado na página 19.

CALLEJAS-CUERVO, M.; ALVAREZ, J. C.; ÁLVAREZ, D. Capture and analysis of biomechanical signals with inertial and magnetic sensors as support in physical rehabilitation processes. In: IEEE. *2016 IEEE 13th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*. [S.l.], 2016. Citado na página 16.

CAMERON, J. D. B. . R. F. B. . M. E. Changing the world with a raspberry pi. *Journal of Computing Sciences in Colleges*, 2013. Citado na página 23.

CHEN, P.-z. et al. Real-time human motion capture driven by a wireless sensor network. *International Journal of Computer Games Technology*, Hindawi Publishing Corp., v. 2015, 2015. Citado na página 16.

DIEBEL, J. Representing attitude: Euler angles, unit quaternions, and rotation vectors. *Matrix*, v. 58, 2006. Citado 3 vezes nas páginas 8, 26 e 27.

EL-SHEIMY S. NASSAR, A. N. N. Wavelet de-noising for imu alignment. *IEEE Aerospace and Electronic Systems Magazine*, 2004. Citado 3 vezes nas páginas 8, 18 e 19.

ESCHERT, S. B. . W. H. . T.; DISSELHORST-KLUG, C. Controlling of a ros-based robotic system in accordance to the assist-as-needed principle in end-effector based rehabilitation systems. 2018. Citado na página 34.

FERREIRA, A. B. D. H. Dicionário aurélio da língua portuguesa. *Positivo*, 2010. Citado na página 46.

FONG, D. T. et al. A wireless motion sensing system using adxl mems accelerometers for sports science applications. In: IEEE. *Fifth World Congress on Intelligent Control and Automation (IEEE Cat. No. 04EX788)*. [S.l.], 2004. v. 6. Citado na página 14.

FONG W. T. ; ONG, S. K. . N. A. Y. C. Methods for in-field user calibration of an inertial measurement unit without external equipment. *Measurement Science and Technology*, 2008. Citado na página 18.

FOXLIN, E. Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Computer Graphics and Applications*, 2005. Citado 2 vezes nas páginas 17 e 19.

GARCIA, R.; LUMSDAINE, A. Multiarray: a c++ library for generic programming with arrays. *Software: Practice and Experience*, Wiley Online Library, v. 35, n. 2, 2005. Citado 2 vezes nas páginas 29 e 30.

GASTALDI, L. et al. Technical challenges using magneto-inertial sensors for gait analysis. In: IEEE. *2016 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*. [S.l.], 2016. Citado na página 21.

GOYAL, S. J. . A. V. . L. Raspberry pi based interactive home automation system through e-mail. *2014 International Conference on Reliability Optimization and Information Technology (ICROIT)*, 2014. Citado na página 23.

GRANDELL, L. et al. Why complicate things?: introducing programming in high school using python. In: AUSTRALIAN COMPUTER SOCIETY, INC. *Proceedings of the 8th Australasian Conference on Computing Education-Volume 52*. [S.l.], 2006. Citado na página 28.

GRECU, C.; IORDACHE, C. Portable i2c monitor and debugger. In: IEEE. *2015 IEEE 21st International Symposium for Design and Technology in Electronic Packaging (SIITME)*. [S.l.], 2015. Citado na página 22.

GRUBIN, C. Derivation of the quaternion scheme via the euler axis and angle. *Journal of Spacecraft and Rockets*, v. 7, n. 10, 1970. Citado na página 27.

GYPSY. *Gypsy 7^{TM} Motion Capture System*. [S.l.]: Capture, 2016. Citado na página 17.

HALDER, R. et al. Formal verification of ros-based robotic applications using timed-automata. In: IEEE. *2017 IEEE/ACM 5th International FME Workshop on Formal Methods in Software Engineering (FormaliSE)*. [S.l.], 2017. p. 44–50. Citado na página 33.

HEMAMI, H. A state space model for interconnected rigid bodies. *IEEE Transactions on Automatic Control*, IEEE, v. 27, n. 2, 1982. Citado 2 vezes nas páginas 16 e 27.

HONDORI, H. M.; KHADEMI, M.; LOPES, C. V. Monitoring intake gestures using sensor fusion (microsoft kinect and inertial sensors) for smart home tele-rehab setting. In: *2012 1st Annual IEEE Healthcare Innovation Conference*. [S.l.: s.n.], 2012. Citado na página 16.

IBRAHIM, H. A.; ALY, A. K.; FAR, B. H. A system for vehicle collision and rollover detection. In: IEEE. *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*. [S.l.], 2016. p. 1–6. Citado 2 vezes nas páginas 27 e 28.

INDUSTRIAL, R. About ros. 2009. Citado na página 31.

JACOB, A.; ZAKARIA, W. N. W.; TOMARI, M. R. B. M. Implementation of imu sensor for elbow movement measurement of badminton players. In: IEEE. *2016 2nd IEEE International Symposium on Robotics and Manufacturing Automation (ROMA)*. [S.l.], 2016. Citado na página 19.

JEFIZA, A. et al. Fall detection based on accelerometer and gyroscope using back propagation. In: IEEE. *2017 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*. [S.l.], 2017. p. 1–6. Citado na página 28.

JOSEPH, L. *Robot Operating System (ROS) for Absolute Beginners*. [S.l.: s.n.], 2018. Citado na página 60.

J.RODRIGUEZ-LERA LEANDRO GOMES, P. Z. A. N. A. S. A.-M. S. F. Emotional robots for coaching: Motivating physical rehabilitation using emotional robots. *Emotional Robots for Coaching*, 2018. Citado na página 33.

JUDY, J. W. Microelectromechanical systems (mems): fabrication, design and applications. *Smart Materials and Structures*, 2001. Citado na página 18.

KULIC, D.; VENTURE, G.; NAKAMURA, Y. Detecting changes in motion characteristics during sports training. In: IEEE. *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. [S.l.], 2009. Citado na página 16.

LABORATóRIO DE ROBóTICA (LARA). *DEMO SOFTWARE JUST FOR EXPERIMENT PURPOER IN A NONCOMERTIAL ACTIVITY - pid.ino*. Disponível em: <https://github.com/raphabraccialli/planta_1dof/blob/master/pid/pid.ino>. Acesso em: 04 jun. 2019. Citado na página 41.

LAMBRECHT, J. M.; KIRSCH, R. F. Miniature low-power inertial sensors: promising technology for implantable motion capture systems. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, IEEE, v. 22, n. 6, 2014. Citado na página 18.

LAVER, K. Virtual reality for stroke rehabilitation. *John Wiley & Sons, Ltd.*, 2011. Citado na página 33.

LEENS, F. An introduction to i2c and spi protocols. *IEEE Instrumentation & Measurement Magazine*, 2009. Citado 2 vezes nas páginas 8 e 22.

LIANZA, S. Medicina de reabilitação. 4 edicao. guanabara koogan. Rio de Janeiro, 2007. Citado na página 35.

LIBERTY. Polhemus innovation in motion. 2016. Citado na página 17.

LIN, . D. K. J. F. S. Human pose recovery using wireless inertial measurement units. *Physiological Measurement*, 2012. Citado na página 17.

MADGWICK, S. O. H. An efficient orientation filter for inertial and inertial/magnetic sensor arrays. 2010. Citado na página 17.

MAZZARI, V. What is ros? *Robotics tutorials*, 2016. Citado na página 32.

MILLMAN, K. J.; AIVAZIS, M. Python for scientists and engineers. *Computing in Science & Engineering*, IEEE, v. 13, n. 2, 2011. Citado na página 29.

MILOSEVIC, B.; FARELLA, E. Wearable inertial sensor for jump performance analysis. In: ACM. *Proceedings of the 2015 workshop on Wearable Systems and Applications*. [S.l.], 2015. Citado na página 16.

MORENO, R. A. M. . G. A. B. . W. R. A. D. . E. D. Raspbian vs ubuntu mate - um paralelo do desempenho na raspberry pi. *XVII Simpósio em Sistemas Computacionais de Alto Desempenho*, 2016. Citado na página 23.

MRTIJN/TIJNDAGAMER. *The communication over I2C between a Raspberry Pi and a MPU-6050 Gyroscope/Accelerometer*. Disponível em: <https://github.com/Tijndagamer/mpu6050/blob/master/mpu6050/mpu6050.py>. Acesso em: 02 jun. 2019. Citado na página 37.

NIU X. ; EL-SHEIMY, N. Development of a low-cost mems imu/gps navigation system for land vehicles using auxiliary velocity updates in the body frame proc. *ION GNSS*, 2005. Citado na página 19.

O'FLYNN, B. et al. Metrics for monitoring patients progress in a rehabilitation context: a case study based on wearable inertial sensors. IARIA, 2018. Citado na página 35.

OLIPHANT, T. E. Python for scientific computing. *Computing in Science & Engineering*, IEEE, v. 9, n. 3, 2007. Citado 2 vezes nas páginas 29 e 31.

ONG, Z. C. et al. Development of an economic wireless human motion analysis device for quantitative assessment of human body joint. *Measurement*, Elsevier, v. 115, 2018. Citado na página 14.

ONG, Z. C. et al. Development of an economic wireless human motion analysis device for quantitative assessment of human body joint. *Measurement*, Elsevier, v. 115, 2018. Citado na página 20.

OSRF. *Writing a Simple Publisher and Subscriber*. Disponível em: <http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber>. Acesso em: 09 maio. 2019. Citado na página 37.

OUDJIDA, A. K. et al. Fpga implementation of i 2 c & spi protocols: A comparative study. In: IEEE. *2009 16th IEEE International Conference on Electronics, Circuits and Systems-(ICECS 2009)*. [S.l.], 2009. Citado na página 23.

PAUL, R. P. *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*. [S.l.]: Richard Paul, 1981. Citado na página 26.

PEREZ, J. G. Introducción a ros en raspberry pi. 2017. Citado na página 32.

PINTO, E. et al. A health and usage monitoring system for ros-based service robots. In: IEEE. *2015 10th International Symposium on Mechatronics and its Applications (ISMA)*. [S.l.], 2015. p. 1–6. Citado na página 35.

PIO, R. Euler angle transformations. In: IEEE. *IEEE Transactions on Automatic Control*. [S.l.], 1996. Citado na página 24.

POLLA, . L. F. F. D. L. Ferroelectric thin films in micro-electromechanical systems applications. *MRS Bulletin*, 1996. Citado na página 18.

POPESCU VLAD CIOBANU, A. G. F. D. D. P. N. Predefined recovery exercises system for after stroke hand rehabilitation. *International Conference and Exposition on Electrical and Power Engineering*, 2016. Citado na página 34.

QUALISYS. Swedish motion capture company. 2016. Citado na página 17.

QUIGLEY, B. G. M.; SMART, W. D. *Programming Robots with ROS*. [S.l.]: O'Reilly Media, Inc, 2010. Citado na página 32.

QUIGLEY, M. et al. Ros: an open-source robot operating system. In: KOBE, JAPAN. *ICRA workshop on open source software*. [S.l.], 2009. v. 3, n. 3.2. Citado na página 35.

ROS MSG AND SRV. *Creating a ROS msg and srv*. Disponível em: <http://wiki.ros.org/pt_BR/ROS/Tutorials/CreatingMsgAndSrv>. Acesso em: 12 maio 2019. Citado na página 37.

ROS NODES. *Understanding ROS Nodes*. Disponível em: <http://wiki.ros.org/ROS/Tutorials/UnderstandingNodes>. Acesso em: 09 março 2019. Citado na página 37.

ROSSUM, G. V. et al. Computer programming for everybody. *Proposal to the Corporation for National Research initiatives*, 1999. Citado na página 28.

SEO, J.-Y.; NOH, Y.-H.; JEONG, D.-U. Implementation of emg data-based rehabilitation assistance system. In: IEEE. *2018 International Conference on Electronics, Information, and Communication (ICEIC)*. [S.l.], 2018. Citado na página 20.

SMITH, J. Ubuntu. 2015. Citado na página 24.

SOLDOVIERI, F.; SOLIMENE, R.; PRISCO, G. A multiarray tomographic approach for through-wall imaging. *IEEE Transactions on Geoscience and Remote Sensing*, IEEE, v. 46, n. 4, 2008. Citado na página 30.

SZEKACS, A.; SZAKAILL, T.; HEGYKOZI, Z. Realising the spi communication in a multiprocessor system. In: IEEE. *2007 5th International Symposium on Intelligent Systems and Informatics*. [S.l.], 2007. Citado na página 22.

TITTERTON D. ; WESTON, J. Strapdown inertial navigation technology. *Institution of Electrical Engineers*, 1997. Citado na página 19.

TSARDOULIAS E. ; MITKAS, A. Robotic frameworks, architectures and middleware comparison. *ITI - Information Technologies Institute*, Grécia, 1998. Citado na página 32.

UBUNTU, M. T. Ubuntu mate. 2018. Citado na página 24.

VENKATRAMAN, S. et al. Wireless inertial sensors for monitoring animal behavior. In: IEEE. *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. [S.l.], 2007. Citado na página 16.

VILLENEUVE, E. et al. Signal quality and compactness of a dual-accelerometer system for gyro-free human motion analysis. *IEEE Sensors Journal*, IEEE, v. 16, n. 16, 2016. Citado na página 20.

WALT, S. V. D.; COLBERT, S. C.; VAROQUAUX, G. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, IEEE Computer Society, v. 13, n. 2, p. 22, 2011. Citado na página 31.

WILLEMSEN, A. T. M.; FRIGO, C.; BOOM, H. B. Lower extremity angle measurement with accelerometers-error and sensitivity analysis. *IEEE transactions on biomedical engineering*, IEEE, v. 38, n. 12, 1991. Citado na página 19.

WREN, T. A. et al. Efficacy of clinical gait analysis: A systematic review. *Gait & posture*, Elsevier, v. 34, n. 2, 2011. Citado na página 21.

XSENS. The leading innovator in 3d motion tracking technology. 2016. Citado na página 17.

XU, B. The design and implementation of a motion capture system based on mems sensors and zigbee network [m.s. thesis]. University of Electronic Science and Technology, 2013. Citado na página 16.

YIN, Z. et al. A novel wireless motion sensor for analyzing golf swing. In: IEEE. *SENSORS, 2013 IEEE*. [S.l.], 2013. p. 1–4. Citado na página 20.

YU, S.-H. et al. Virtual reality survival first person shooting game. In: IEEE. *2017 International Automatic Control Conference (CACS)*. [S.l.], 2017. Citado na página 57.

Apêndices

APÊNDICE A – Hardware

Iniciou-se os processos realizando-se a verificação da ativação do endereço do sensor. Foi confirmado o respectivo endereçamento, 0x68. A conexão do hardware do primeiro sensor MPU-6050 ligada a raspberry pi 3, foi realizada da seguinte forma:

Pinagem	
Raspberry Pi 3	**Módulo GY-521**
DC POWER - 3.3V	VCC
GPIO02	$SDA1 - I^2C$
GPIO03	$SCL1 - I^2C$
GND	GND

Tabela 3 – Pinagem conexão MPU-6050 e Raspberry PI 3.

Após concluir com êxito os testes do primeiro Nó “publisher”, conectamos o segundo módulo GY-521 a raspberry cumprindo a pinagem descrita abaixo na tabela 4.

Pinagem		
Raspberry Pi 3	**Módulo GY-521 (1)**	**Módulo GY-521 (2)**
DC POWER - 3.3V	VCC	VCC
GPIO02	$SDA1 - I^2C$	$SDA1 - I^2C$
GPIO03	$SCL1 - I^2C$	$SCL1 - I^2C$
GND	GND	GND
-	AD0 - VCC	AD0 - GND

Tabela 4 – Pinagem conexão dos dois módulos GY-521.

Para conectar vários dispositivos ao barramento I2C, o pino AD0 no módulo GY-521 foi projetado para alterar seu endereço e assim, estabelecer comunicação (YU et al., 2017). Desta maneira, o AD0 opera de forma diretamente dependente do nível lógico configurado. Encontramos alguns percalços durante as primeiras tentativas de endereçamento do segundo MPU-6050, nas quais não havíamos ligado corretamente os respectivos pinos AD0 em VCC e GND. O primeiro MPU 6050 foi conectado em nível lógico alto e o segundo em nível baixo.Posteriormente, habilitou-se a comunicação I2C. O endereço equivalente aos sensores são 0x68 e 0x69, como é possível observar na figura 16.

```
carol@carol-rpi:~$ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 69 -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
carol@carol-rpi:~$
```

Figura 16 – Endereçamento via I2C dos dois MPU's

Na figura 17 podemos observar a conexão final dos módulos MPU6050 com a Raspberry Pi, bem como a interconexão entre os respectivos pinos dos componentes.

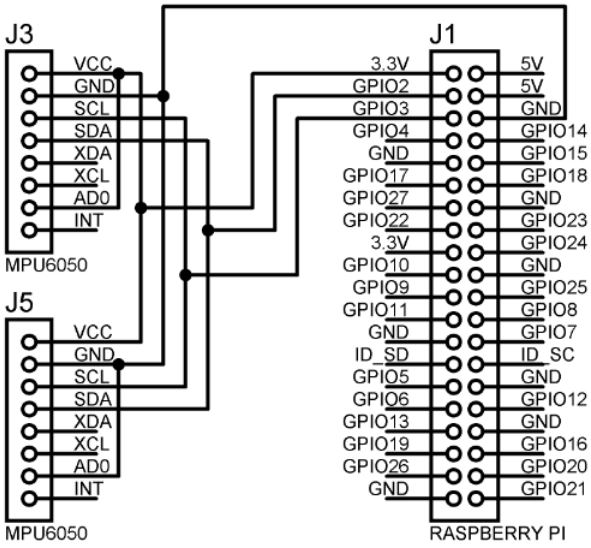

Figura 17 – Descrição da posição de todos os pinos do circuito do protótipo. Pinagem realizada no Software Proteus (Autoria própria).

Anexos

ANEXO A – Instalação da ROS

O ROS pode ser instalado em PCs/desktops e em placas embarcadas. Primeiramente ao iniciar a instalação, um cartão de memória(micro SDCard) é requerido, foi utilizado para o projeto um cartão SanDisk 16 GB. O sistema operacional Ubuntu MATE foi selecionado principalmente pela compatibilidade de suas funções com a versão Kinetic Kame e RPi 3B.

O download do software Ubuntu MATE específico para Raspberry Pi 2 ou 3, foi realizado diretamente na página oficial do sistema, disponível em (http://ubuntu-mate.org/download/), versão 16.04.2(Xenial). Depois da realização do último procedimento citado, com o micro cartão SD inserido no computador, é necessário baixar o programa SD Memory Card Formatter.Para formatar o cartão introduzido: <https://www.sdcard.org/downloads/formatter_4/>.

Posteriormente, selecione o pacote .zip do sistema operacional com o botão direito, e opte por extraí-lo utilizando o software 7-zip, https://www.7-zip.org/, após à conclusão da extração, o arquivo.img requer o download do Win32 Disk Imager , projetado para gravar uma imagem de disco bruta em um dispositivo removível , pode ser acessado em https://sourceforge.net/projects/win32diskimager/, para isso, é preciso selecionar e carregar o arquivo.img no Win32, e clicar em Writer, assim ele irá gravar o arquivo.img no cartão SD. Após todo o processo realizado, o SD pode ser retirado do computador e inserido na plataforma Raspberry Pi 3. Depois do cabo HDMI ser conectado juntamente aos periféricos: teclado e mouse e ao ser alimentada com uma fonte de 5V e 3A, o sistema dá um boot inicial e abre a interface clara e intuitiva do Ubuntu MATE.

Para download do pacote ROS após a configuração completa do Ubuntu MATE, deve-se abrir o terminal MATE para ter acesso ao prompt de comando, seguindo o tutorial fornecido em <http://wiki.ros.org/kinetic/Installation/Ubuntu>, primordialmente pede-se para configurar o sistema para receber os novo pacotes, digitando a linha de comando:

$ sudo sh -c ’echo "deb (http://packages.ros.org/ros/ubuntu) $(lsb_release -sc) main» /etc/apt/sources.list.d/ros-latest.list’

Em seguida, as chaves de configuração do sistema devem ser baixadas, utilizando:

$ sudo apt-key adv –keyserver hkp://ha.pool.sks-keyservers.net:80 –recv-key

421C365BD9FF1F717815A3895523BAEEB01FA116

Agora, antes de iniciar o download do sistema ROS, é recomendado checar se a versão atual está com todos os pacotes completos, solicitando a atualização dos pacotes Ubuntu. Assim, o sistema listará, e verificará nos pacotes contidos se há alguma atuali-

zação disponível,

$ sudo apt-get update

É possível instalar pacotes ROS individualmente. Porém, optou-se por baixar o mais recomendado, o Desktop-Full Install, é o pacote mais completo, com diversas recursos úteis para os desdobramentos do projeto, como rqt, rviz, robot-generic libraries, 2D/3D simulators, navigation e 2D/3D perception. Leva cerca de, algumas dezena de minutos para realizar o download de todos os recursos disponíveis, e pode ser baixado, através do comando:

$ sudo apt-get instala ros-kinetic-desktop-full

Após a instalação de todos os pacotes, antes de iniciar a utilização da plataforma ROS, é necessário iniciar o Rosdep, que é uma ferramenta de linha de comando para a instalação de dependências do sistema, e é requerido para executar alguns componentes principais no ROS. Por exemplo, um pacote típico do ROS pode pertencer a alguns pacotes dependentes para funcionar corretamente. Rosdep verifica se os pacotes dependentes estão disponíveis, e se não, instala-os automaticamente. É muito recomendado no começo da inicialização do sistema, pois ele irá instalar dependências de códigos úteis para execução de certos códigos do próprio núcleo ROS.Seguindo os comandos abaixo, pode-se iniciá-lo:

$ sudo rosdep init $ rosdep update

Como já citado, a ROS vem com ferramentas e bibliotecas já inclusas, mas para acessá-las deve-se configurar o ambiente requerido. O seguinte comando adiciona uma linha no arquivo .bashrc em sua pasta pessoal, que define o ambiente ROS em todos os novos terminais.É conveniente que as variáveis de ambiente do ROS sejam automaticamente adicionadas à sua sessão bash toda vez que um novo shell for lançado.Ao adicionar as variáveis de ambiente no shell quando o terminal for aberto, os comando do ROS serão reconhecidos.

$ echo "source /opt/ros/kinetic/setup.bash»> /.bashrc

Insira também o próximo comando, com a finalidade de adicionar esse ambiente no terminal atual.

$ source /.bashrc

De acordo com os comandos já instalados, os pacotes essenciais já podem ser executados. Entretanto, existem dezenas de ferramentas que são disponibilizadas separadamente. Rosinstall é um exemplo, é frequentemente muito utilizada, permite realizar o download com facilidade de muitas árvores de pacotes ROS com um comando. Numa situação hipotética onde um robô precisa configurar uma pilha com milhares de pacotes ao mesmo tempo, Rosinstall seria de grande valia, instalando com apenas um comando todos os processos de uma única vez(JOSEPH, 2018).

$ sudo apt-get install python-rosinstall python-rosinstall-generator python-wstool build-essential

Caso queira conferir se a sua instalação está correta, basta usar:

$ rosversion -d

Finalmente, ROS Kinetic Kame instalado e pronto para utilização.

ANEXO B – Simulação

Passo-a-passo da simulação do sistema desenvolvido em funcionamento.

Primeiro terminal aberto no prompt de comando inicia a ROS com o comando ROSCORE, como citado anteriormente.

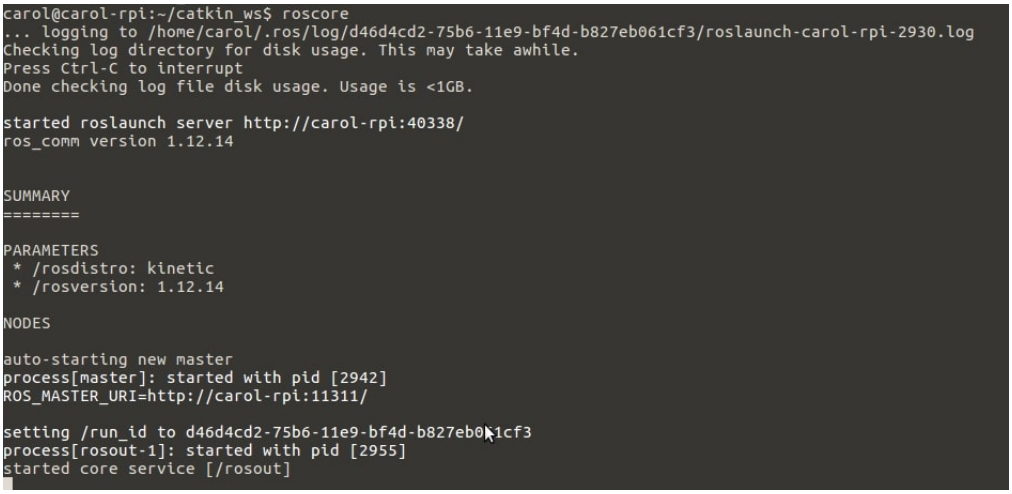

Figura 18 – Comando roscore.

O segundo terminal aberto, publica os ângulos pitch e roll do primeiro sensor.

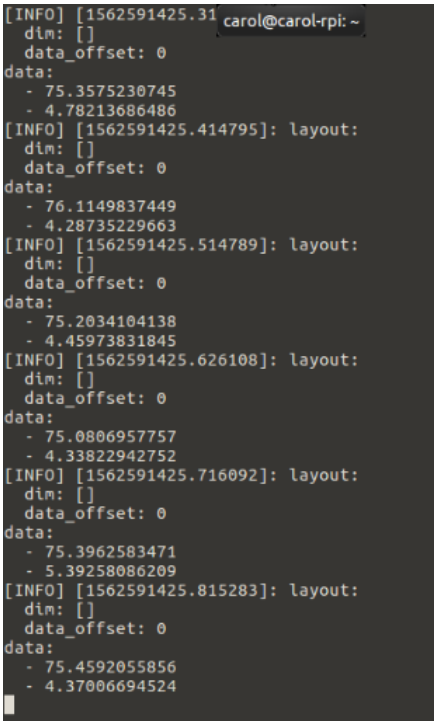

Figura 19 – Comando rosrun sensor 1.

O terceiro terminal aberto, publica os ângulos pitch e roll do segundo sensor.

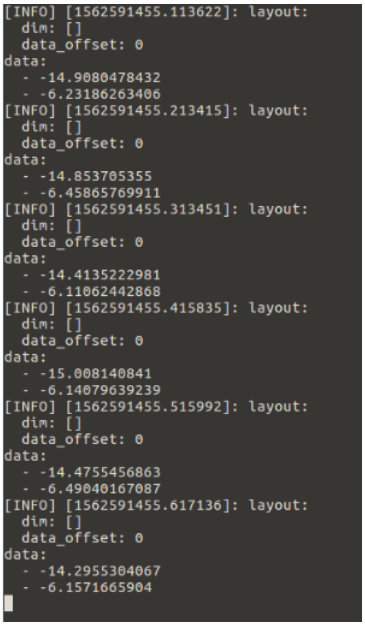

Figura 20 – Comando rosrun sensor 2.

O quarto terminal aberto, publica os ângulos pitch e roll do primeiro sensor dentro da ROS.

O quinto terminal aberto, publica os ângulos pitch e roll do segundo sensor dentro da ROS. E finalmente, o sexto terminal, publica a subtração dos ângulos recebidos pelos dois sensores. Podem ser observados nas seguintes figuras 21 e 22.

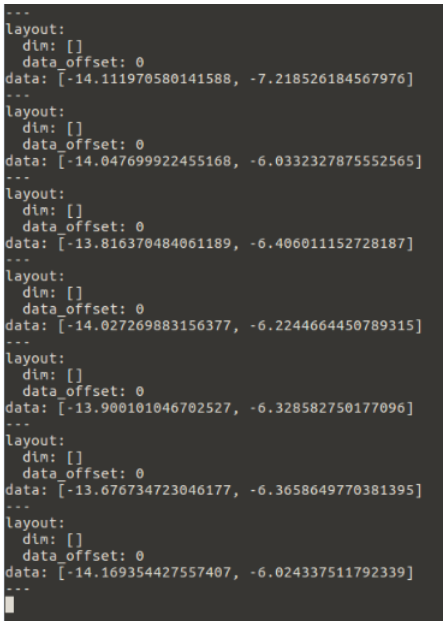

Figura 22 – Comando rostopic sensor 2.

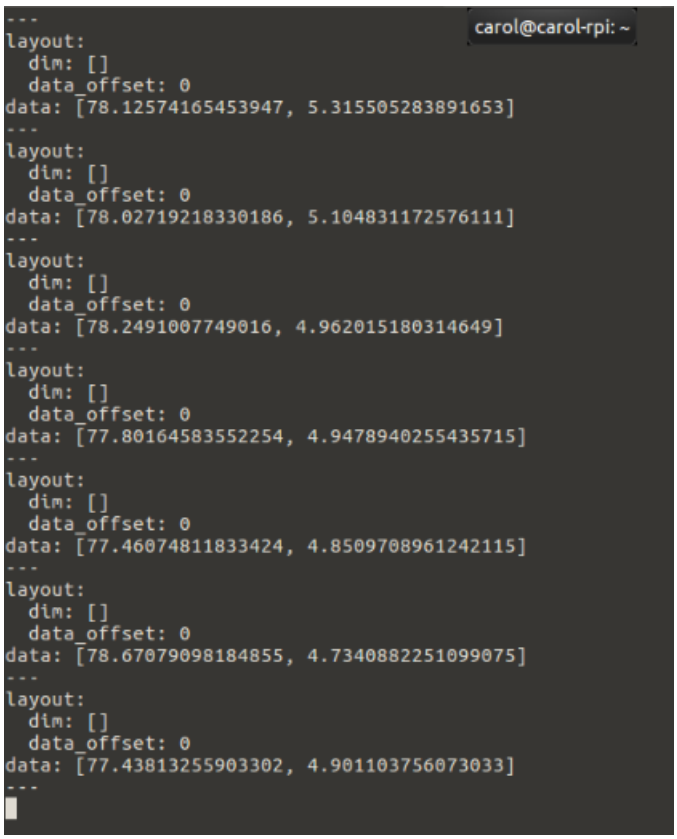

Figura 21 – Comando rostopic sensor 1.

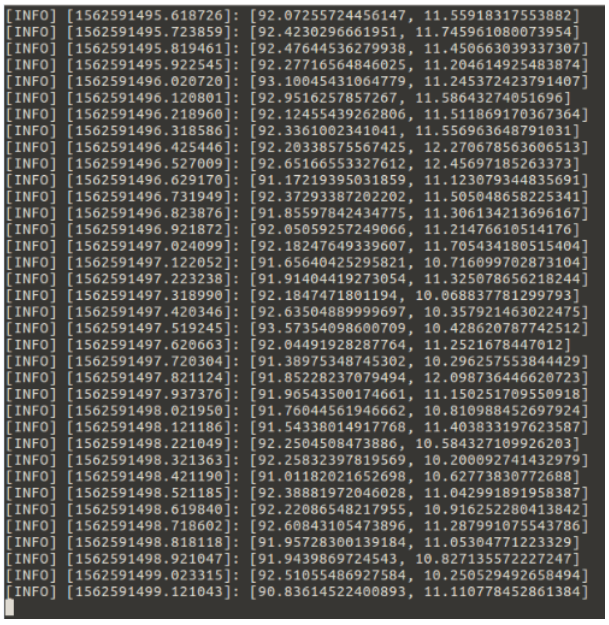

Figura 23 – Publicação da Subtração dos ângulos.

ANEXO C – Especificações Acelerômetro

Especificações do sensor MPU6050 – Acelerômetro

VDD = 2.375V-3.46V, VLOGIC (MPU-6050 only) = 1.8V±5% or VDD, T_A = 25°C

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
ACCELEROMETER SENSITIVITY						
Full-Scale Range	AFS_SEL=0		±2		*g*	
	AFS_SEL=1		±4		*g*	
	AFS_SEL=2		±8		*g*	
	AFS_SEL=3		±16		*g*	
ADC Word Length	Output in two's complement format		16		bits	
Sensitivity Scale Factor	AFS_SEL=0		16,384		LSB/*g*	
	AFS_SEL=1		8,192		LSB/*g*	
	AFS_SEL=2		4,096		LSB/*g*	
	AFS_SEL=3		2,048		LSB/*g*	
Initial Calibration Tolerance			±3		%	
Sensitivity Change vs. Temperature	AFS_SEL=0, -40°C to +85°C		±0.02		%/°C	
Nonlinearity	Best Fit Straight Line		0.5		%	
Cross-Axis Sensitivity			±2		%	
ZERO-G OUTPUT						
Initial Calibration Tolerance	X and Y axes		±50		mg	1
	Z axis		±80		mg	
Zero-G Level Change vs. Temperature	X and Y axes, 0°C to +70°C		±35			
	Z axis, 0°C to +70°C		±60		mg	
SELF TEST RESPONSE						
Relative	Change from factory trim	-14		14	%	2
NOISE PERFORMANCE						
Power Spectral Density	@10Hz, AFS_SEL=0 & ODR=1kHz		400		µg/√Hz	
LOW PASS FILTER RESPONSE						
	Programmable Range	5		260	Hz	
OUTPUT DATA RATE						
	Programmable Range	4		1,000	Hz	
INTELLIGENCE FUNCTION INCREMENT			32		mg/LSB	

1. Typical zero-g initial calibration tolerance value after MSL3 preconditioning
2. Please refer to the following document for further information on Self-Test: *MPU-6000/MPU-6050 Register Map and Descriptions*

ANEXO D – Especificações Giroscópio

Especificações do sensor MPU6050 – Giroscópio

VDD = 2.375V-3.46V, VLOGIC (MPU-6050 only) = 1.8V±5% or VDD, T_A = 25°C

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
GYROSCOPE SENSITIVITY						
Full-Scale Range	FS_SEL=0		±250		°/s	
	FS_SEL=1		±500		°/s	
	FS_SEL=2		±1000		°/s	
	FS_SEL=3		±2000		°/s	
Gyroscope ADC Word Length			16		bits	
Sensitivity Scale Factor	FS_SEL=0		131		LSB/(°/s)	
	FS_SEL=1		65.5		LSB/(°/s)	
	FS_SEL=2		32.8		LSB/(°/s)	
	FS_SEL=3		16.4		LSB/(°/s)	
Sensitivity Scale Factor Tolerance	25°C	-3		+3	%	
Sensitivity Scale Factor Variation Over Temperature			±2		%	
Nonlinearity	Best fit straight line; 25°C		0.2		%	
Cross-Axis Sensitivity			±2		%	
GYROSCOPE ZERO-RATE OUTPUT (ZRO)						
Initial ZRO Tolerance	25°C		±20		°/s	
ZRO Variation Over Temperature	-40°C to +85°C		±20		°/s	
Power-Supply Sensitivity (1-10Hz)	Sine wave, 100mVpp; VDD=2.5V		0.2		°/s	
Power-Supply Sensitivity (10 - 250Hz)	Sine wave, 100mVpp; VDD=2.5V		0.2		°/s	
Power-Supply Sensitivity (250Hz - 100kHz)	Sine wave, 100mVpp; VDD=2.5V		4		°/s	
Linear Acceleration Sensitivity	Static		0.1		°/s/*g*	
SELF-TEST RESPONSE						
Relative	Change from factory trim	-14		14	%	1
GYROSCOPE NOISE PERFORMANCE	**FS_SEL=0**					
Total RMS Noise	DLPFCFG=2 (100Hz)		0.05		°/s-rms	
Low-frequency RMS noise	Bandwidth 1Hz to10Hz		0.033		°/s-rms	
Rate Noise Spectral Density	At 10Hz		0.005		°/s/ √ Hz	
GYROSCOPE MECHANICAL FREQUENCIES						
X-Axis		30	33	36	kHz	
Y-Axis		27	30	33	kHz	
Z-Axis		24	27	30	kHz	
LOW PASS FILTER RESPONSE						
	Programmable Range	5		256	Hz	
OUTPUT DATA RATE						
	Programmable	4		8,000	Hz	
GYROSCOPE START-UP TIME	**DLPFCFG=0**					
ZRO Settling (from power-on)	to ±1°/s of Final		30		ms	

1. Please refer to the following document for further information on Self-Test: *MPU-6000/MPU-6050 Register Map and Descriptions*